AFATL-TR-78-81

Further Development Of The EPIC-3 Computer Program For Three-Dimensional Analysis Of Intense Impulsive Loading

HONEYWELL, INC.
DEFENSE SYSTEMS DIVISION
600 SECOND STREET NORTHEAST
HOPKINS, MINNESOTA 55343

JULY 1978

FINAL REPORT FOR PERIOD APRIL 1977-APRIL 1978

Approved for public release; distribution unlimited

Air Force Armament Laboratory
AIR FORCE SYSTEMS COMMAND ★ UNITED STATES AIR FORCE ★ EGLIN AIR FORCE BASE, FLORIDA

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (WHEN DATA ENTERED)

19 REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER 18 AFATL-TR-78-81	2. GOV'T ACCESSION NUMBER	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (AND SUBTITLE) 6 FURTHER DEVELOPMENT OF THE EPIC-3 COMPUTER PROGRAM FOR THREE-DIMENSIONAL ANALYSIS OF INTENSE IMPULSIVE LOADING.		5. TYPE OF REPORT/PERIOD COVERED 9 Final Report, Apr[redacted]77 - Apr[redacted]78,
		6. PERFORMING ORG. REPORT NUMBER 14 47324
7. AUTHOR(S) 10 Gordon R. / Johnson, Dwight D. / Colby, Daniel J. / Navrick		8. CONTRACT OR GRANT NUMBER(S) 15 F08635-77-C-0121 new
9. PERFORMING ORGANIZATIONS NAME/ADDRESS Honeywell Inc. Defense Systems Division, 600 Second Street Northeast, Hopkins, Minnesota 55343		10. PROGRAM ELEMENT PROJECT TASK AREA & WORK UNIT NUMBERS JON 16 2307-E2-20 17 E2, Program Element: 61602F
11. CONTROLLING OFFICE NAME/ADDRESS Air Force Armament Laboratory, Armament Development and Test Center, Eglin Air Force Base, Florida 32542		12. REPORT DATE 11 July[redacted]78
		13. NUMBER OF PAGES 116
14. MONITORING AGENCY NAME/ADDRESS (IF DIFFERENT FROM CONT. OFF.) 12 116p.		15. SECURITY CLASSIFICATION (OF THIS REPORT) UNCLASSIFIED
		15a. DECLASSIFICATION DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (OF THIS REPORT)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (OF THE ABSTRACT ENTERED IN BLOCK 20, IF DIFFERENT FROM REPORT)

18. SUPPLEMENTARY NOTES

Available in DDC.

19. KEY WORDS (CONTINUE ON REVERSE SIDE IF NECESSARY AND IDENTIFY BY BLOCK NUMBER)

Impact	Elastic-plastic
Penetration	Hydrodynamic
Wave Propagation	Three-Dimensional
Finite Element	Lagrangian

20. ABSTRACT (CONTINUE ON REVERSE SIDE IF NECESSARY AND IDENTIFY BY BLOCK NUMBER)

An advanced version of the EPIC-3 computer program is described. The numerical technique is based on a Lagrangian finite element formulation where the equations of motion are integrated directly rather than through the traditional stiffness matrix approach. It is primarily intended for three dimensional analysis of penetration and wave propagation problems resulting from high velocity impact, although other applications are possible. The two most significant improvements over the initial version of the EPIC-3 code, are that problems of unlimited size can be run and that → next page

79 02 09 083

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 55 IS OBSOLETE

393 249 Gu

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (WHEN DATA ENTERED)

20. ABSTRACT (Continued)

the code is programmed to eventually run on fourth generation vector computers. Other improvements include the capability to apply external pressures, reduction of required input, expanded geometry generators, a more general sliding surface capability and expanded plotting capability.

PREFACE

This final report on the development of the advanced EPIC-3 code was prepared by Honeywell Inc., Defense Systems Division, for the Air Force Armament Laboratory, Armament Development and Test Center, Eglin Air Force Base, Florida 32542 under contract F08635-77-C-0121. The period covered by the report is April 1977 to April 1978. The authors, G. R. Johnson, D. D. Colby and D. J. Vavrick, would like to thank Lt. Colonel John J. Osborn, Major Daniel A. Matuska, and Mr. W. H. Cook of Eglin Air Force Base for their assistance in running the sample problems.

This report has been reviewed by the Information Officer (OI) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be available to the general public, including foreign nations.

This technical report has been reviewed and is approved for publication.

FOR THE COMMANDER:

John R Taylor

JOHN R. TAYLOR, Lt. Colonel, USAF
Chief, Munitions Division

(The reverse of this page is blank)

/

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF FIGURES (Concluded)

(The reverse of this page is blank)

SECTION I
INTRODUCTION AND SUMMARY

This report documents an advanced version of the EPIC-3 computer program (Elastic-Plastic Impact Computations in 3 dimensions). It is primarily intended for analysis of three-dimensional wave propagation and penetration problems resulting from high-velocity impact. It is based on an explicit Lagrangian finite-element formulation where the equations of motion are integrated directly rather than through the traditional stiffness matrix approach. This code is an extension of an earlier version of EPIC-3 (Reference 1) with significant improvements which are summarized later in this section.

The EPIC-3 code has material descriptions which include strain hardening, strain rate effects, thermal softening and fracture. Geometry generators are included to quickly generate flat plates, spheres and rods with various nose shapes. It has the capability to include sliding surfaces, and it provides plots of deformed geometry and velocity vectors. In addition, the finite-element computational approach has several specific advantages when compared to the more commonly used finite-difference approach:

- The finite-element formulation does not require an orderly grid and is therefore well-suited to represent complicated geometrical shapes by use of three-dimensional tetrahedron elements.
- The finite-element formulation allows the use of tetrahedron elements which are well-suited to represent severe distortions.

- The tetrahedron elements are in a state of constant strain such that all material in an element behaves uniformly. This allows for an accurate and convenient selection of constant stresses and pressures within the element.
- The finite-element formulation is well-suited for various boundary conditions. Free boundaries require no special accommodations, and restrained boundaries are simply represented by setting the appropriate nodal displacements and velocities equal to zero.

Although the advanced EPIC-3 code has many similarities to the earlier version (Reference 1), there are many specific extensions and improvements. A summary of these follows:

- The better-structured code consists of three separate programs: Preprocessor, Main Routine, and Postprocessor. These three programs are connected by a common restart tape.
- Provided the nodal bandwidth can be core contained, problems of unlimited size can be run by buffering data between disk files and core. For smaller problems, where all nodes can be core contained, an option is provided to bypass this buffering.
- The code is programmed to eventually run on fourth-generation vector computers. It is expected that it will also run faster on the CDC 6600 and 7600 computers since these machines also have some vectorization characteristics.
- Capability is provided to apply external pressures on the triangular faces of the tetrahedron elements. Pressures can also be applied as a function of time.

- Elastic snapback is included to allow slower problems to be run more accurately. Internal energy computations are also improved to better conserve total energy (kinetic plus internal) under all conditions.
- Reduced input is required for sliding surfaces and geometry generators.
- Expanded geometry generators include solid and hollow rods with various nose shapes, solid and hollow spheres, and flat plates with variable expansion factors. There are no limitations on the number of rings in the rods, noses and spheres.
- Expanded sliding-surface capability includes frictional effects. A general search routine is also provided for cases where the top surface of the target is not a single-valued function along lines normal to the top of the undeformed target.
- Expanded plotting capability provides two- and three-dimensional plots of geometry and velocity vectors.

The following sections of this report present the formulation, a description of the computer program, user instructions, and example problems.

SECTION II
FORMULATION

1. COMPUTATIONAL TECHNIQUE

The EPIC-3 computational technique is shown schematically in Figure 1. The first step in the process is to represent the geometry with tetrahedron elements having specific material characteristics. Then the distributed mass is lumped at the nodes (element corners), and initial velocities are assigned to represent the motion at impact. If the problem involves applied pressures, there may be no initial velocities, and the initial conditions are established by the application of the pressures.

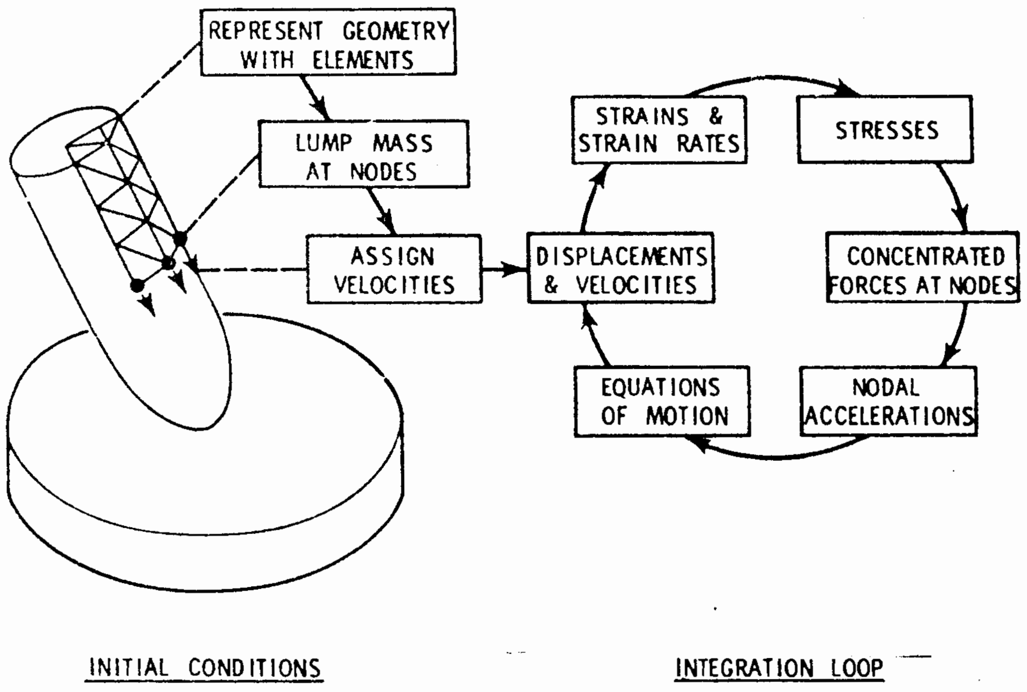


Figure 1. EPIC-3 Computational Technique

After the initial conditions are established, the integration loop begins as shown in Figure 1. The first step is to obtain displacements and velocities of the nodes. If it is assumed the lines connecting the nodes (element edges) remain straight, then the displacements and velocities within the elements must vary linearly. From these displacements and velocities, the strains and strain rates within the elements can be obtained. Since the strains and strain rates are derivatives of linear displacement and velocity functions within the elements, the resulting strains and strain rates are constant within the elements.

The stresses in the elements are determined from the strains, strain rates, internal energies and material properties. Since the strains and strain rates are constant within the elements, the stresses are also constant. The stresses are obtained by combining elastic or plastic deviator stresses with hydrostatic pressure. The deviator stresses represent the shear strength capability of the material, and the hydrostatic pressure is obtained from the volumetric strain and internal energy of the element. An artificial viscosity is also included to damp out localized oscillations caused by representing continuous media with lumped masses.

After the element stresses are determined, it is necessary to obtain concentrated forces at the nodes. These forces are statically equivalent to the distributed stresses within the element and the applied pressures acting on the faces of the element. They are dependent on the element geometry and the magnitude of the stresses and pressures. When the concentrated forces are applied to the concentrated masses, the nodal accelerations are defined, and the equations of motion are applied to determine new displacements and velocities. The integration loop is then repeated until the time of interest has elapsed.

Another feature of the basic technique is the ability to represent sliding between two surfaces. This is accomplished with a momentum exchange principle which allows for closing, sliding, and separation of the two surfaces. It should be noted that the integration time increment must be properly controlled to prevent numerical instability. This is accomplished by limiting the time increment to a fraction of the time required to travel across the minimum altitude of the element at the sound velocity of the material.

2. ELEMENT GEOMETRY

A typical tetrahedron element is shown in Figure 2. It is geometrically defined by nodes i, j, m, and p. The formulation is based on nodes i, j, and m being positioned in a counterclockwise manner when viewed from node p. The mass of each element is $V_o\rho_o$ where V_o and ρ_o are the initial volume and density of the element. This mass is equally distributed to concentrated masses at the four nodes such that the total mass at node i, M_i, is equal to one-fourth the mass of all elements which contain that node. The coordinates of node i are designated x_i, y_i, z_i, and the corresponding velocities are designated $\dot{u}_i$, $\dot{v}_i$, $\dot{w}_i$.

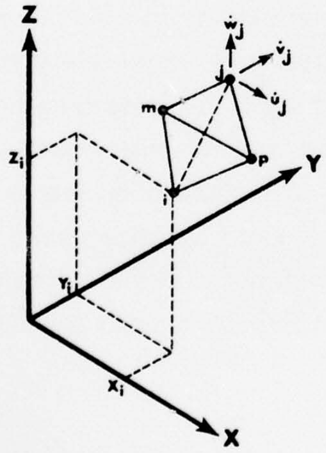


Figure 2. Tetrahedron Element Geometry

3. STRAINS AND STRAIN RATES

The incremental strains which occur during each cycle of integration are obtained by multiplying the strain rates by the integration time increment. The strain rates are obtained from the current geometry of the element and the velocities of the nodes. If it is assumed the velocities vary linearly between the nodes, the x, y, and z velocities ($\dot{u}$, $\dot{v}$, $\dot{w}$) within each element can be expressed as

$$\dot{u} = \alpha_1 + \alpha_2 x + \alpha_3 y + \alpha_4 z \tag{1}$$

$$\dot{v} = \alpha_5 + \alpha_6 x + \alpha_7 y + \alpha_8 z \tag{2}$$

$$\dot{w} = \alpha_9 + \alpha_{10} x + \alpha_{11} y + \alpha_{12} z \tag{3}$$

where $\alpha_1 \ . \ . \ . \ \alpha_{12}$ are geometry- and velocity-dependent constants for each element. It is possible to solve for $\alpha_1 \ . \ . \ . \ \alpha_4$ by substituting the x velocities and coordinates of the four nodes into Equation (1). This gives four equations and four unknowns so that the constants ($\alpha_1 \ . \ . \ . \ \alpha_4$) can be evaluated.

Equation (1) can then be expressed in terms of element geometry and nodal velocities.

$$\begin{aligned} \dot{u} = \frac{1}{6V} \Big[& (a_i + b_i x + c_i y + d_i z)\, \dot{u}_i \\ + & (a_j + b_j x + c_j y + d_j z)\, \dot{u}_j \\ + & (a_m + b_m x + c_m y + d_m z)\, \dot{u}_m \\ + & (a_p + b_p x + c_p y + d_p z)\, \dot{u}_p \Big] \end{aligned} \tag{4}$$

where the volume is

$$V = \frac{1}{6} \begin{vmatrix} 1 & x_i & y_i & z_i \\ 1 & x_j & y_j & z_j \\ 1 & x_m & y_m & z_m \\ 1 & x_p & y_p & z_p \end{vmatrix} \tag{5}$$

and the other geometry-dependent constants are

$$a_i = \begin{vmatrix} x_j & y_j & z_j \\ x_m & y_m & z_m \\ x_p & y_p & z_p \end{vmatrix} \tag{6a}$$

$$b_i = - \begin{vmatrix} 1 & y_j & z_j \\ 1 & y_m & z_m \\ 1 & y_p & z_p \end{vmatrix} \tag{6b}$$

$$c_i = \begin{vmatrix} 1 & x_j & z_j \\ 1 & x_m & z_m \\ 1 & x_p & z_p \end{vmatrix} \tag{6c}$$

$$d_i = - \begin{vmatrix} 1 & x_j & y_j \\ 1 & x_m & y_m \\ 1 & x_p & y_p \end{vmatrix} \tag{6d}$$

The remaining geometry-dependent constants (a_j, b_j, c_j, d_j, etc.) are obtained by a systematic interchange of signs and subscripts. The y and z velocities in Equations (2) and (3) are obtained in a similar manner and are identical to Equation (1) except the x velocities at the four nodes are replaced by the y and z velocities.

After the velocities are obtained, it is possible to determine the normal strain rates ($\dot{\epsilon}_x$, $\dot{\epsilon}_y$, $\dot{\epsilon}_z$), the shear strain rates ($\dot{\gamma}_{xy}$, $\dot{\gamma}_{xz}$, $\dot{\gamma}_{yz}$) and the spin rates (ω_x, ω_y, ω_z) of the element.

$$\dot{\epsilon}_x = \frac{\partial \dot{u}}{\partial x} \tag{7}$$

$$\dot{\epsilon}_y = \frac{\partial \dot{v}}{\partial y} \tag{8}$$

$$\dot{\epsilon}_z = \frac{\partial \dot{w}}{\partial z} \tag{9}$$

$$\dot{\gamma}_{xy} = \frac{\partial \dot{u}}{\partial y} + \frac{\partial \dot{v}}{\partial x} \tag{10}$$

$$\dot{\gamma}_{xz} = \frac{\partial \dot{u}}{\partial z} + \frac{\partial \dot{w}}{\partial x} \tag{11}$$

$$\dot{\gamma}_{yz} = \frac{\partial \dot{v}}{\partial z} + \frac{\partial \dot{w}}{\partial y} \tag{12}$$

$$\omega_x = \frac{1}{2}\left(\frac{\partial \dot{w}}{\partial y} - \frac{\partial \dot{v}}{\partial z}\right) \tag{13}$$

$$\omega_y = \frac{1}{2}\left(\frac{\partial \dot{u}}{\partial z} - \frac{\partial \dot{w}}{\partial x}\right) \tag{14}$$

$$\omega_z = \frac{1}{2}\left(\frac{\partial \dot{v}}{\partial x} - \frac{\partial \dot{u}}{\partial y}\right) \tag{15}$$

It can be seen that Equations (7) through (15) are derivatives of linear functions and therefore give constant values within the element. The volumetric strain is also constant within the element and is expressed as $\epsilon_v = V/V_o - 1$, where V and V_o represent the current and initial volumes of the element. It should also be noted that subsequent computations will involve deviator strain rates ($\dot{e}_x$, $\dot{e}_y$, $\dot{e}_z$) which are readily obtained from the normal strain rates ($\dot{\epsilon}_x$, $\dot{\epsilon}_y$, $\dot{\epsilon}_z$) by subtracting the average normal strain rate.

4. STRESSES AND PRESSURES

The stresses in the elements are determined from the strains, strain rates, internal energies and material properties (Reference 2). The three normal stresses (σ_x, σ_y, σ_z) are expressed in terms of deviator stresses (s_x, s_y, s_z), hydrostatic pressure, P, and artificial viscosity, Q:

$$\sigma_x = s_x - (P + Q) \tag{16}$$

$$\sigma_y = s_y - (P + Q) \tag{17}$$

$$\sigma_z = s_z - (P + Q) \tag{18}$$

Trial values of the deviator stresses and shear stresses at time = t + Δt are

$$s_x^{t + \Delta t} = s_x^t + 2G\dot{e}_x\Delta t + \Delta s_x \tag{19}$$

$$s_y^{t + \Delta t} = s_y^t + 2G\dot{e}_y\Delta t + \Delta s_y \tag{20}$$

$$s_z^{t + \Delta t} = s_z^t + 2G\dot{e}_z\Delta t + \Delta s_z \tag{21}$$

$$\tau_{xy}^{t + \Delta t} = \tau_{xy}^t + G\dot{\gamma}_{xy}\Delta t + \Delta\tau_{xy} \tag{22}$$

$$\tau_{xz}^{t+\Delta t} = \tau_{xz}^{t} + G\dot{\gamma}_{xz}\Delta t + \Delta\tau_{xz} \qquad (23)$$

$$\tau_{yz}^{t+\Delta t} = \tau_{yz}^{t} + G\dot{\gamma}_{yz}\Delta t + \Delta\tau_{yz} \qquad (24)$$

In Equation (19) the first term (s_x^t) is the normal stress at the previous time and the second term ($2G\dot{e}_x\Delta t$) is the incremental stress due to the incremental strain ($\dot{e}_x\Delta t$) during that time increment, where G is the elastic shear modulus and Δt is the integration time increment. The third term (Δs_x) is due to shear stresses from the previous time increment, which now act as normal stresses due to the new orientation of the element caused by an incremental rotation ($\omega_y\Delta t$, $\omega_z\Delta t$) during the time increment. The remaining normal stresses and shear stresses have a similar form.

The correction terms for element rotations are

$$\Delta s_x = 2\,(\omega_y\tau_{xz}^t - \omega_z\tau_{xy}^t)\,\Delta t \qquad (25)$$

$$\Delta s_y = 2\,(\omega_z\tau_{xy}^t - \omega_x\tau_{yz}^t)\,\Delta t \qquad (26)$$

$$\Delta s_z = 2\,(\omega_x\tau_{yz}^t - \omega_y\tau_{xz}^t)\,\Delta t \qquad (27)$$

$$\Delta\tau_{xy} = [\omega_z\,(s_x^t - s_y^t) + \omega_y\tau_{yz}^t - \omega_x\tau_{xz}^t]\,\Delta t \qquad (28)$$

$$\Delta\tau_{xz} = [\omega_y\,(s_z^t - s_x^t) + \omega_x\tau_{xy}^t - \omega_z\tau_{yz}^t]\,\Delta t \qquad (29)$$

$$\Delta\tau_{yz} = [\omega_x\,(s_y^t - s_z^t) + \omega_z\tau_{xz}^t - \omega_y\tau_{xy}^t]\,\Delta t \qquad (30)$$

It should be recalled that Equations (19) through (24) represent trial values of the stresses, and they may need to be reduced if they violate the Von Mises yield criterion. An equivalent stress is given by

$$\bar{\sigma} = \sqrt{\frac{3}{2}(s_x^2 + s_y^2 + s_z^2) + 3(\tau_{xy}^2 + \tau_{xz}^2 + \tau_{yz}^2)} \tag{31}$$

If $\bar{\sigma}$ is not greater than the equivalent tensile strength of the material, $\bar{S}$, (which is discussed later in this section), the final deviator and shear stresses are as given in Equations (19) through (24). If $\bar{\sigma}$ is greater than $\bar{S}$, then the stresses in Equations (19) through (24) should be multiplied by the factor $(\bar{S}/\bar{\sigma})$. When the reduced deviator and shear stresses are put into Equation (31), the result is always $\bar{\sigma} = \bar{S}$.

During plastic flow it is sometimes necessary to determine the equivalent plastic strain for strain hardening effects on the strength of the material or to determine if the material has failed. The first step in this process is to adjust the strain rates to plastic strain rates by substracting out the elastic portion of the strain rates:

$$\dot{e}_x^p = \dot{e}_x - \frac{s_x^{t+\Delta t} - s_x^t - \Delta s_x}{2G\Delta t} \tag{32}$$

$$\dot{e}_y^p = \dot{e}_y - \frac{s_y^{t+\Delta t} - s_y^t - \Delta s_y}{2G\Delta t} \tag{33}$$

$$\dot{e}_z^p = \dot{e}_z - \frac{s_z^{t+\Delta t} - s_z^t - \Delta s_z}{2G\Delta t} \tag{34}$$

$$\dot{\gamma}_{xy}^p = \dot{\gamma}_{xy} - \frac{\tau_{xy}^{t+\Delta t} - \tau_{xy}^t - \Delta\tau_{xy}}{G\Delta t} \tag{35}$$

$$\dot{\gamma}_{xz}^p = \dot{\gamma}_{xz} - \frac{\tau_{xz}^{t+\Delta t} - \tau_{xz}^t - \Delta\tau_{xz}}{G\Delta t} \tag{36}$$

$$\dot{\gamma}^p_{yz} = \dot{\gamma}_{yz} - \frac{\tau^{t+\Delta t}_{yz} - \tau^t_{yz} - \Delta\tau_{yz}}{G\Delta t} \tag{37}$$

The equivalent plastic strain rate is expressed as

$$\dot{\bar{\epsilon}}_p = \sqrt{\frac{2}{9}\left[(\dot{e}^p_x - \dot{e}^p_y)^2 + (\dot{e}^p_x - \dot{e}^p_z)^2 + (\dot{e}^p_y - \dot{e}^p_z)^2 + \frac{3}{2}(\dot{\gamma}^{p\,2}_{xy} + \dot{\gamma}^{p\,2}_{xz} + \dot{\gamma}^{p\,2}_{yz})\right]} \tag{38}$$

The equivalent plastic strain, $\bar{\epsilon}_p$, is then obtained by integrating $\dot{\bar{\epsilon}}_p$ with respect to time:

$$\bar{\epsilon}_p^{t+\Delta t} = \bar{\epsilon}_p^t + \dot{\bar{\epsilon}}_p \Delta t \tag{39}$$

The equivalent tensile strength of the material may be dependent on many factors, including strain, strain rate, pressure and temperature. It is well known that many materials behave differently under dynamic impact than under static testing conditions. With few exceptions, however, a precise definition of material behavior under dynamic conditions is not available. There is also much to be learned about fracture characteristics under these dynamic conditions. The current version of EPIC-3 allows the equivalent tensile strength to be determined from

$$\bar{S} = S_{\bar{\epsilon}_p}\,[1 + C_1 \log(\dot{\bar{\epsilon}})]\,[1 + C_2\bar{P} + C_3\bar{P}^2]\,[C_4 + C_5 T] \tag{40}$$

In this formulation, $S_{\bar{\epsilon}_p}$ is generally taken to be the static stress, which is dependent on the equivalent plastic strain of Equation (39). The three bracketed terms allow the static stress to be altered, based on strain rate, pressure and temperature. If the constants are $C_1 = C_2 = C_3 = C_5 = 0$ and $C_4 = 1.0$, then $\bar{S} = S_{\bar{\epsilon}_p}$, which is the strain-dependent static stress. The first bracketed term can increase the strength due to the equivalent strain rate, $\dot{\bar{\epsilon}}$, which is similar to that described in Equation (38), but includes the elastic portion of the strain rates. In the second bracketed term, $\overline{P} = P + Q$, so the effect of this term is to increase the strength due to pressure. The final bracketed term includes the temperature, T, of the element, which is obtained from

$$T = T_o + \frac{E_s}{C_s \rho_o} \tag{41}$$

where T_o is the initial temperature of the element, E_s is the internal energy per original unit volume (defined later), C_s is the specific heat and ρ_o is the initial density.

The material is assumed to fracture if a specified value of the equivalent plastic strain, $\bar{\epsilon}_p$ (Equation 39) or the volumetric strain, $\epsilon_v = V/V_o - 1$, is exceeded. This is a very approximate failure model which should be expanded when better models become available. When the failure criterion has been met (exceeded value of $\bar{\epsilon}_p$ or ϵ_v), the equivalent tensile stress is set to zero so no shear stresses can be developed in the failed element. Likewise, no tensile stresses are allowed to develop. The net result is that a failed element tends to act like a liquid inasmuch as it can develop only hydrostatic compression with no shear or tensile stresses. Another option is also available which sets all stresses (including the pressure) equal to zero.

The hydrostatic pressure is dependent on the volumetric strain and the internal energy in the element. The EPIC-3 code uses the Mie-Grüneisen equation of state (Reference 3) in the general form $P = P_v + \Gamma E_s (1 + \mu)$, with the complete expression given by

$$P = (K_1\mu + K_2\mu^2 + K_3\mu^3)\,(1 - \frac{\Gamma\mu}{2}) + \Gamma E_s\,(1 + \mu) \tag{42}$$

where $\mu = V_o/V-1$, K_1, K_2, and K_3 are material-dependent constants and Γ is the Grüneisen coefficient.

Since the pressure can be significantly affected by the internal energy E_s, it is desirable to solve the pressure and energy equations simultaneously. This gives

$$E_s^{t+\Delta t} = \frac{E_s^t - .5\,[(P+Q)^t + Q^{t+\Delta t} + P_v^{t+\Delta t}]\,\dot{\epsilon}_v \Delta t + \Delta E_d}{1 + .5\Gamma\,(1+\mu)\,\dot{\epsilon}_v \Delta t} \tag{43}$$

where $\dot{\epsilon}_v$ is the volumetric strain rate and ΔE_d is the internal energy generated by the deviator and shear stresses during the previous cycle.

$$\Delta E_d = (\bar{s}_x\dot{e}_x + \bar{s}_y\dot{e}_y + \bar{s}_z\dot{e}_z + \bar{\tau}_{xy}\dot{\gamma}_{xy} + \bar{\tau}_{xz}\dot{\gamma}_{xz} + \bar{\tau}_{yz}\dot{\gamma}_{yz}) \quad (\bar{V}/V_o)\,\Delta t \tag{44}$$

The bars on the deviator and shear stresses and the volume represent averages of these values at times t and t + Δt. After the total internal energy at time t + Δt has been determined from Equation (43), the pressure at time t + Δt is determined from Equation (42).

The artificial viscosity is combined with the normal stresses to damp out localized oscillations of the concentrated masses. It tends to eliminate spurious oscillations which would otherwise occur for wave propagation problems. This technique was originally proposed by Von Neumann and Richtmyer (Reference 4) and has been expanded for use in various computer codes (Reference 5). It is expressed in terms of linear and quadratic components and is applied only when the volumetric strain rate is negative.

$$Q = C_L \rho c_s h \, |\dot{\epsilon}_v| + C_Q \rho h^2 \, (\dot{\epsilon}_v)^2 \quad \text{for } \dot{\epsilon}_v < 0$$

$$Q = 0 \quad \text{for } \dot{\epsilon}_v \geq 0 \tag{45}$$

where c_s is the sound velocity of the material and h is the minimum altitude of the tetrahedron. C_L and C_Q are the linear and quadratic dimensionless coefficients.

The minimum altitude of the tetrahedron can readily be obtained from the previously calculated geometry-dependent constants. The altitude from node i to the plane defined by the other three nodes of the tetrahedron is

$$h_i = \frac{6V}{\sqrt{b_i^2 + c_i^2 + d_i^2}} \tag{46}$$

The other altitudes (h_j, h_m, h_p) are obtained by appropriately changing the subscripts of the geometry-dependent constants.

The sound velocities (Reference 6) are obtained from

$$c_s^2 = \frac{1}{\rho_o} [K_1 \, (1 - \Gamma\mu) + K_2 \, (2\mu - 1.5\Gamma\mu^2) + K_3 \, (3\mu^2 - 2\,\Gamma\mu^3) + \Gamma E_s + \Gamma\overline{P}/(1 + \mu)] \tag{47}$$

The sound velocities are also used for determination of the integration time increment.

5. CONCENTRATED FORCES

After the element stresses are determined, the concentrated nodal forces can be obtained. These forces are statically equivalent to the distributed stresses within the element and the applied pressures acting on the faces of the element. They are dependent on the displaced element geometry and the magnitude of the stresses and pressures. The forces in the x, y and z directions at node i of an element are given by

$$F_x^i = \frac{1}{6}[F_t (b_j P_j + b_m P_m + b_p P_p) - (b_i \sigma_x + c_i \tau_{xy} + d_i \tau_{xz})] \quad (48)$$

$$F_y^i = \frac{1}{6}[F_t (c_j P_j + c_m P_m + c_p P_p) - (c_i \sigma_y + b_i \tau_{xy} + d_i \tau_{yz})] \quad (49)$$

$$F_z^i = \frac{1}{6}[F_t (d_j P_j + d_m P_m + d_p P_p) - (d_i \sigma_z + c_i \tau_{yz} + b_i \tau_{xz})] \quad (50)$$

The geometry-dependent constants (b_i, c_i, d_i) are again identical to those used for calculation of the strain rates and altitudes. The terms in the first sets of parentheses represent the applied pressures where P_j, P_m, and P_p are the external pressures applied to the triangular faces opposite to nodes j, m, and p respectively. The pressures are all multiplied by a common time-dependent factor, F_t. The terms in the second sets of parentheses represent the effect of the internal element stresses. The forces at the other nodes are readily obtained by changing subscripts. The net forces at node i ($\bar{F}_x^i$, $\bar{F}_y^i$, $\bar{F}_z^i$) are the sum of the node i forces from each element which includes that node.

6. EQUATIONS OF MOTION

The equations of motion can be numerically integrated by assuming a constant velocity for each time increment. The acceleration of node i in the x direction at time = t is

$$\ddot{u}_i^t = \frac{\overline{F}_x^i}{M_i} \tag{51}$$

The new constant velocity for the next time increment is

$$\dot{u}_i^{t+} = \dot{u}_i^{t-} + \ddot{u}_i^t \overline{\Delta t} \tag{52}$$

where $\dot{u}_i^{t-}$ is the constant velocity for the previous time increment and $\overline{\Delta t}$ is the average of the two integration time increments about time = t. Finally, the new displacement at time = t + Δt is

$$u_i^{t+\Delta t} = u_i^t + \dot{u}_i^{t+} \Delta t \tag{53}$$

The equations of motion for the y and z directions have a similar form.

The integration time increment used for the equations of motion is given by

$$\Delta t = C_t \left[\frac{h}{\sqrt{g^2} + \sqrt{g^2 + c_s^2}} \right] \tag{54}$$

where $g^2 = C_Q Q/\rho$, h is the previously determined minimum altitude, and c_s is the sound velocity. The constant, C_t, must be less than unity to ensure that Δt is always less than the time required to travel across the shortest dimension of the tetrahedron at the sound velocity of the material.

Use of a time increment significantly larger than that specified by Equation (54) will lead to numerical instability (Reference 5). The EPIC-3 program restricts the time increment from increasing more than 10 percent per cycle.

7. SLIDING SURFACES

It is sometimes necessary to allow for sliding to occur between two surfaces. A summary of the important steps for the sliding-surface technique follows:

- Identify a master sliding surface defined by an orderly arrangement of master nodes.
- Identify slave nodes which may slide along the master surface.
- For each integration time increment, apply the equations of motion to both the master nodes and the slave nodes in the usual manner.
- For each slave node, find the triangular plane (defined by three master nodes) whose projection contains that slave node.
- Check to determine if there is interference between the slave node and the master surface (the triangular plane defined by three master nodes).
- If there is interference, place the slave node on the master surface in a direction normal to the appropriate master triangular plane.

- Calculate the momentum change (impulse) to the slave node caused by placing it on the master surface; then transfer this lost momentum to the three surrounding master nodes in a direction normal to the master surface.
- Calculate the frictional impulse developed between the slave node and the three master nodes. The impulse is applied in the plane of the master surface and in a direction opposing the relative motion.

The first step in the process is to define the master surface. It is generally desirable to define this surface with an orderly arrangement of nodes since a systematic search is required on this surface. An example of this is shown at upper left in Figure 3 where a flat plate is represented by a finite-element model of nodes and elements. The z axis points vertically upward and the x and y axes are in a horizontal plane. Although the basic principles involved in this technique are independent of the orientation of the axes, the search technique is simplified if the z coordinates of the master surface are limited to a single-valued function of x and y (i. e., any vertical line parallel to the z axis must not pass through the master surface at more than one point). This specialized case will be described first, and the specific characteristics of the generalized case will be described later.

For the specialized case, the slave surface is above the master surface relative to the z axis. Due to the symmetry of the plate about a plane containing the x and z axes, only one-half the plate is considered as shown. The EPIC-3 code uses a master surface defined by M rows of nodes, with each row containing N nodes. Looking downward from the positive z axis, the second row of nodes from N + 1 to 2N is to the left of the first row of nodes going from 1 to N. Likewise, the third row is to the left of the second row, etc. For the specific case shown in Figure 3, N = 17 and M = 4, for a total of 68 master nodes.

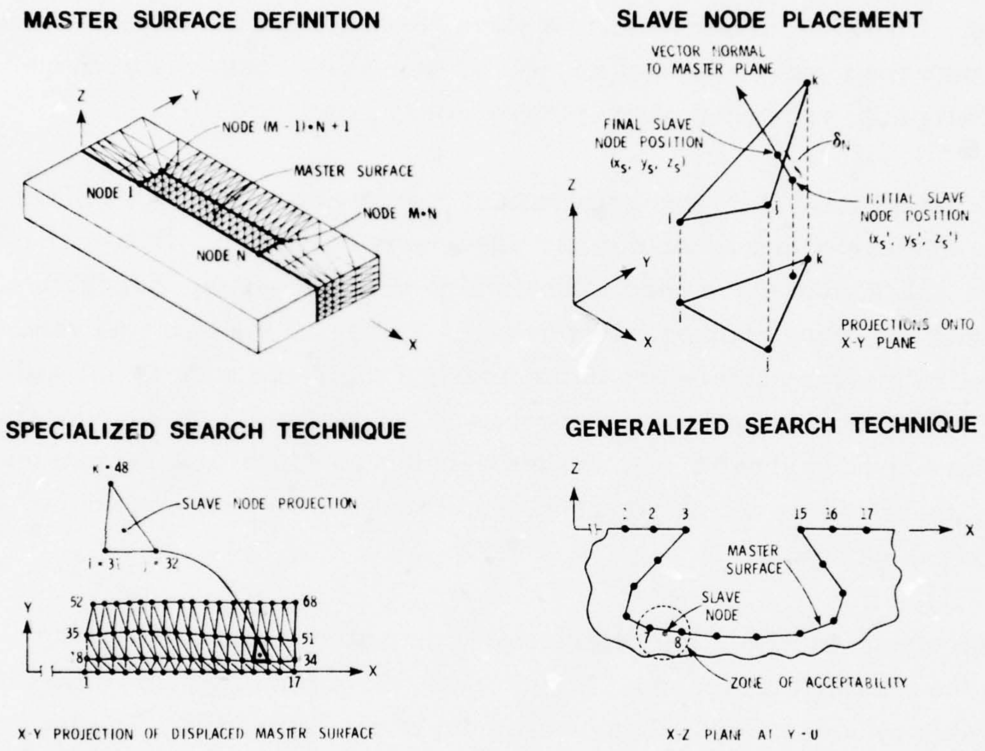

MASTER SURFACE DEFINITION
Z
Y
NODE (M - 1)•N + 1
MASTER SURFACE
NODE 1
NODE M•N
NODE N
X
SLAVE NODE PLACEMENT
VECTOR NORMAL TO MASTER PLANE
FINAL SLAVE NODE POSITION
(x_s, y_s, z_s)
δ_N
INITIAL SLAVE NODE POSITION
(x_s', y_s', z_s')
PROJECTIONS ONTO X-Y PLANE
SPECIALIZED SEARCH TECHNIQUE
k = 48
SLAVE NODE PROJECTION
i = 31
j = 32
X-Y PROJECTION OF DISPLACED MASTER SURFACE
GENERALIZED SEARCH TECHNIQUE
MASTER SURFACE
SLAVE NODE
ZONE OF ACCEPTABILITY
X-Z PLANE AT Y = 0

Figure 3. Sliding Surface Procedure

It is also necessary to identify slave nodes for the other sliding surface. Since these nodes do not require an orderly arrangement, it is usually convenient to designate the more geometrically complex surface as the slave surface. It is also desirable to restrict the slave node spacing from becoming significantly greater than the master node spacing, because this could introduce localized deformations in the master surface at the slave node locations. Likewise, the mass of the slave nodes should not be significantly greater than the mass of the master nodes, since this could result in unrealistically high velocities of the master nodes at impact.

For each integration loop, the equations of motion are applied to both the master and slave nodes in the manner previously described. It is then necessary to check each slave node to determine if it has passed through the master surface, thus causing interference. To begin, the extreme values of the entire master surface are determined; if the slave node is not within this region, there can be no interference. If, for instance, the z coordinate of the slave node is greater than the maximum z coordinate of the master surface, there can be no interference, and the check for that particular slave node is complete.

If the slave node is within the confines of the master surface, a search through the master surface must be initiated. It is for this search that it is desirable to have an orderly arrangement of master nodes. The lower left portion of Figure 3 shows the projection of an arbitrarily displaced master surface on the x-y plane. The triangular grid is consistent with the triangular faces of the tetrahedron elements. Since z is a single-valued function, there is no overlapping of triangles on the x-y projection. The objective of the search is to determine which triangle contains the x-y projection for each slave node. The search begins by considering the triangles between the first two rows of nodes. With reference to the lower left portion of Figure 3, the first triangle is defined by nodes 1, 2, 18, the second by nodes 2, 19, 18, and the third by nodes 2, 3, 19, etc. If the slave node projection is not found by the last triangle in the row (nodes 17, 34, 33), the pattern is repeated in the second row of triangles beginning with the triangle defined by nodes 18, 19, 35.

An efficient way to check if the slave node projection is within a specific triangle is to determine the distance from the slave node to each of the lines defining the three sides of the triangle. The distance to line i-j in the x-y plane is given by

$$\delta_{xy} = \frac{\beta_1 x'_s + \beta_2 y'_s + \beta_3}{\sqrt{\beta_1^2 + \beta_2^2}} \tag{55}$$

where $\beta_1 = y_i - y_j$, $\beta_2 = x_j - x_i$ and $\beta_3 = x_i y_j - x_j y_i$. The coordinates of master node i are x_i, y_i and z_i, and the coordinates of the slave node are x'_s, y'_s, and z'_s. If master node j is counterclockwise from node i, when looking downward from the positive z axis, and δ_{xy} is positive for all three lines of the triangle, then the slave node is contained within the triangle. This is also illustrated in Figure 3 where the slave node is included in the triangle defined by nodes 31, 32 and 48.

After the appropriate triangle is identified, it is necessary to determine if there is interference. The equation of a plane through nodes i, j and k (taken counterclockwise, when looking downward from the positive z axis) is

$$Ax + By + Cz + D = 0 \tag{56}$$

where

$$A = \begin{vmatrix} y_i & z_i & 1 \\ y_j & z_j & 1 \\ y_k & z_k & 1 \end{vmatrix} \qquad (57a)$$

$$B = - \begin{vmatrix} x_i & z_i & 1 \\ x_j & z_j & 1 \\ x_k & z_k & 1 \end{vmatrix} \qquad (57b)$$

$$C = \begin{vmatrix} x_i & y_i & 1 \\ x_j & y_j & 1 \\ x_k & y_k & 1 \end{vmatrix} \qquad (57c)$$

$$D = - \begin{vmatrix} x_i & y_i & z_i \\ x_j & y_j & z_j \\ x_k & y_k & z_k \end{vmatrix} \qquad (57d)$$

In the upper right portion of Figure 3, a vector is shown which passes through the initial slave node position and is normal to the triangular plane. The three direction cosines of this vector are readily obtained by dividing A, B and C by $\sqrt{A^2 + B^2 + C^2}$. Subsequent formulations will consider the constants (A, B, C, D) to be redefined such that A, B, C represent the direction cosines. The normal distance between the slave node and the master plane is

$$\delta_N = - (Ax'_s + By'_s + Cz'_s + D) \qquad (58)$$

where x'_s, y'_s and z'_s are the coordinates of the slave node. If δ_N is positive, there is interference and the slave node is placed onto the master surface in a direction normal to the master surface. The resulting final coordinates of the slave node are $x_s = x'_s + \Delta x$, $y_s = y'_s + \Delta y$ and $z_s = z'_s + \Delta z$, where Δx, Δy and Δz are obtained by multiplying δ_N by the appropriate direction cosines of the normal vector.

After geometric compatibility has been achieved by placing the slave node on the master surface, it is necessary to adjust the equations of motion to account for this change. Since the geometric changes are made only in the direction normal to the master plane, the affected equations of motion are altered only in this normal direction. It should be noted that frictional effects alter the equations of motion in the plane of the master surface, but these will be considered later.

In the process of moving the slave node to the master surface, an effective velocity change is imposed on the slave node. This loss of velocity in the normal direction is $\Delta V_N = \delta_N / \Delta t$, where Δt is the integration time increment for the previous cycle. This velocity change also gives a momentum change along the normal direction equal to $M_s \Delta V_N$, where M_s is the mass of the slave node. This momentum is then transferred to the three surrounding master nodes. If R_i, R_j and R_k represent the fractions of the momentum to be transferred to the three nodes, a summation of the translational and rotational momenta of the three nodes gives the following equations:

$$R_i + R_j + R_k = 1 \tag{59}$$

$$R_i (x_i - x_s) + R_j (x_j - x_s) + R_k (x_k - x_s) = 0 \tag{60}$$

$$R_i (y_i - y_s) + R_j (y_j - y_s) + R_k (y_k - y_s) = 0 \tag{61}$$

These three equations and three unknowns define the appropriate momentum distribution to the three nodes. This momentum is applied to the master nodes in the form of instantaneous velocity changes, parallel to the normal vector but in the opposite direction. For instance, the normal direction velocity change to node i is

$$\Delta V_i = R_i M_s \Delta V_N / M_i \tag{62}$$

where M_i is the mass of master node i and M_s is the mass of the slave node. As was done for the displacements, the specific velocity changes in the x, y and z directions are obtained by multiplying ΔV_i by the appropriate direction cosines.

It is sometimes desirable to include frictional effects in the analysis. This effect is constrained to the plane of the master surface and the direction opposes the relative motion. The net magnitude of the frictional velocity change to the slave node is proportional to the normal velocity change to the slave node, or

$$\Delta V_P = f_s \Delta V_N \tag{63}$$

where f_s is the coefficient of friction.

To apply this velocity change in the direction of the relative velocities it is necessary to determine the velocity of the master surface at the slave node location. The procedure is similar to that used in Equations (1) through (4) which determined the velocity distribution within a tetrahedron. For this case, however, the velocity is determined within a two-dimensional triangle since it can be defined as a function of the x and y coordinates. The resulting master surface velocity in the x direction, at the slave node position, is

$$\dot{u}_m = q_i \dot{u}_i + q_j \dot{u}_j + q_k \dot{u}_k \tag{64}$$

The other velocity components ($\dot{v}_m$ and $\dot{w}_m$) have a similar form to Equation (64), and the geometry constants (q_i, q_j, q_k), have the form

$$q_i = \frac{1}{2A_{xy}} [x_j y_k - x_k y_j + (y_j - y_k) x_s + (x_k - x_j) y_s] \tag{65}$$

where A_{xy} is the projected cross-sectional area of the triangular master surface on the x-y plane, and the other terms represent the coordinates of the master and slave nodes.

Now that the velocities of both the master surface and slave node are defined, it is necessary to determine the components of these velocities which are in the plane of the master surface. This is accomplished by subtracting the velocity components normal to the master plane, from the total velocity components. These velocity components for the slave node are expressed as

$$\dot{u}_s^P = \dot{u}_s - A\, V_s^N \tag{66}$$

$$\dot{v}_s^P = \dot{v}_s - B\, V_s^N \tag{67}$$

$$\dot{w}_s^P = \dot{w}_s - C\, V_s^N \tag{68}$$

where the slave node velocity normal to the master surface is

$$V_s^N = A\dot{u}_s + B\dot{v}_s + C\dot{w}_s \tag{69}$$

The in-plane velocities of the master surface ($\dot{u}_m^P$, $\dot{v}_m^P$, $\dot{w}_m^P$) are obtained in a similar manner.

The relative velocity components ($\dot{u}^P_{rel}$, $\dot{v}^P_{rel}$, $\dot{w}^P_{rel}$) are then obtained in the form of

$$\dot{u}^P_{rel} = \dot{u}^P_m - \dot{u}^P_s \tag{70}$$

Finally, the friction induced velocity changes ($\Delta\dot{u}^P_s$, $\Delta\dot{v}^P_s$, $\Delta\dot{w}^P_s$) have the form

$$\Delta\dot{u}^P_s = \left(\frac{\dot{u}^P_{rel}}{\sqrt{(\dot{u}^P_{rel})^2 + (\dot{v}^P_{rel})^2 + (\dot{w}^P_{rel})^2}} \right) \Delta V_P \tag{71}$$

The momentum change of the master nodes, due to the frictional force, gives in-plane velocity changes to master node i ($\Delta\dot{u}^P_i$, $\Delta\dot{v}^P_i$, $\Delta\dot{w}^P_i$) of the form

$$\Delta\dot{u}^P_i = -R_i \left(\frac{\dot{u}^P_{rel}}{\sqrt{(\dot{u}^P_{rel})^2 + (\dot{v}^P_{rel})^2 + (\dot{w}^P_{rel})^2}} \right) \left(\frac{M_s}{M_i} \right) \Delta V_P \tag{72}$$

This completes the description of the sliding surface procedure for the specialized search technique shown in Figure 3. The necessity for a generalized search technique is shown in the lower right portion of the same figure where the master surface is not a single-valued function. A slave node is shown slightly below the master triangle containing nodes 7 and 8, and it should properly be placed on this triangle. If the specialized search technique would be used for this condition, however, the slave node would be placed on the triangle containing nodes 2 and 3 since it is also contained in the x-y projection of this triangle.

To alleviate this problem, the generalized search technique selects the master surface triangle which is closest to the slave node. This selected triangle must also be within a zone of acceptability which is dependent on the dimensions of the selected triangular surface. If it is within this zone a check for interference is made with Equation (58). Placement of the slave node onto the master surface and the adjustments of the nodal velocities are done in a similar manner to that which was done for the specialized formulation. In some cases, however, the master triangle must be defined in terms of the x-z or y-z coordinate system, depending on the orientation of the triangle.

The specialized search routine is computationally faster than the generalized search routine since it generally does not search through every triangle on the master surface; if the slave node is beyond the extremes of the master surface, the routine does not even begin the search, and if it does search, it stops when the proper triangle is located. For either routine the translational momenta are absolutely conserved in all three directions. Small errors are introduced into the center of gravity positions when the slave node is moved to the master surface, since there is no corresponding movement of the master nodes, which receive only instantaneous velocity changes. This center of gravity error also introduces small errors into the rotational momenta.

SECTION III
COMPUTER PROGRAM DESCRIPTION

1. BACKGROUND

This section describes the important characteristics of the computer program. The EPIC-3 software was developed in FORTRAN on a Honeywell series 6000 large-scale digital computer. The program is quite portable; only minor changes were required to make it operational on the Eglin Air Force Base CDC 6600 computer. Two major design goals were to develop the program for efficient execution on fourth-generation vector computers and to have the capability to provide solutions for problems of unlimited size.

Even though the advanced EPIC-3 code is a descendant of an earlier version, it was totally redesigned and rewritten. The top-down, hierarchical design approach resulted in a structured collection of functional modules. The software was implemented with an emphasis on clarity and readability. With few exceptions, strict coding conventions and a limited set of coding constructs were employed. The design documentation, written as comment lines, has been embedded into the source code for each module. Vectorization guidelines were applied to major computational loops, whenever possible, to enhance execution speed on computers that have vector-processing capability. Extensive use of files for intermediate storage was required to provide solutions for large problems which cannot be core contained.

The EPIC-3 Postprocessor provides the user with a flexible plot capability to display two- or three-dimensional views of the initial and deformed geometry as well as plots of velocity vectors. The three-dimensional capability is significant because any perspective is available, and hidden lines can be optionally removed in the geometry plots.

2. PROGRAM STRUCTURE

The EPIC-3 software is currently implemented on an H6080 computer. It utilizes the H6080 system sort package and standard CALCOMP Plot calls in addition to intrinsic functions supplied with its FORTRAN complier. The software is also operational on the Eglin AFB CDC 6600 computer, which uses a Stromberg-Carlson Plotter.

As shown in Figure 4, EPIC-3 has three separate programs which communicate by means of one restart file format. The Preprocessor (PREP) sets up the initial problem data, writing it on the restart file. The Main Routine (MAIN) reads this restart file or one created by a previous Main Routine run, and then it processes additional time cycles, outputting to a new restart file at user-specified intervals. The Postprocessor (POST) obtains its data from restart files created by either the Main Routine or the Preprocessor.

The three programs were functionally decomposed with a top-down approach to design. The resultant program structures and module functions are presented as Figures 5 through 7. The data handling, which is common to all EPIC-3 programs, is facilitated by the multi-use service routines listed in Figure 8. A summary of all EPIC-3 subroutines, with their associated common blocks and service routines, is provided in Figure 9.

3. VECTORIZATION GUIDELINES

A requirement for EPIC-3 is that it will eventually be used on vector computers such as the CRAY, and it must therefore be written to be compatible with these machines. Compilers on vector computers such as the CRAY analyze the innermost loops of a program and attempt to use vector instructions which exploit the advanced hardware and software features of the computer. Since no special forms are required for vector operations, the

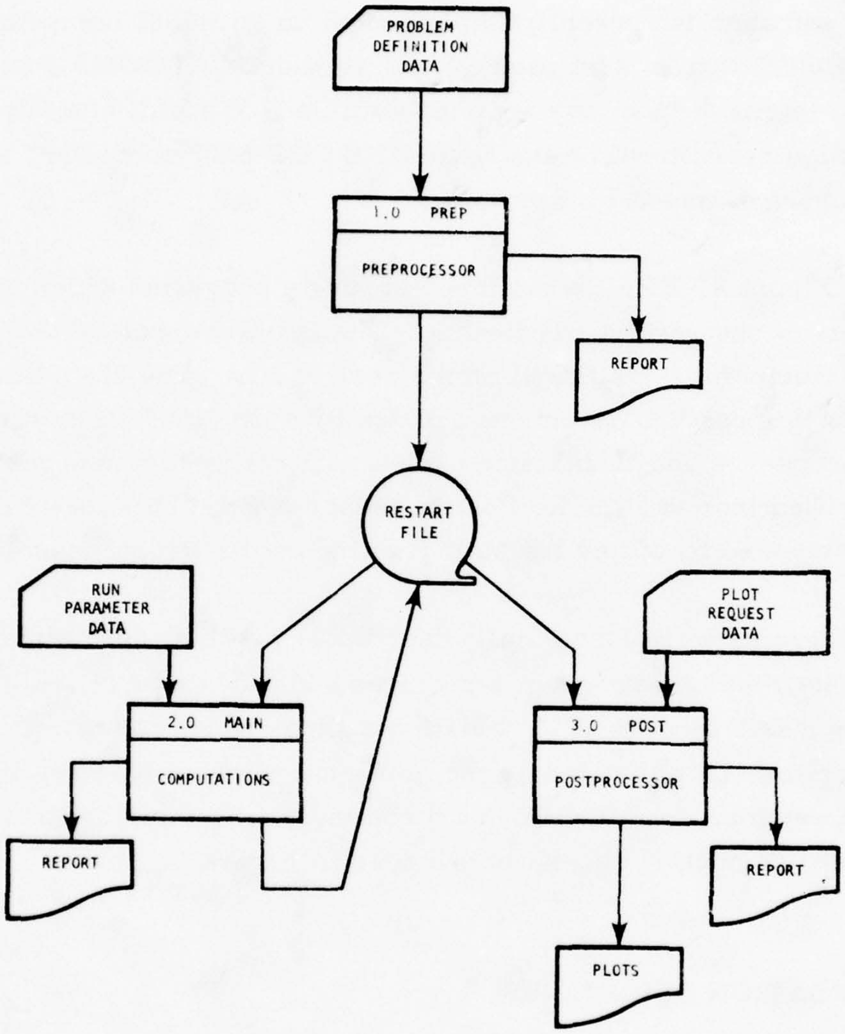
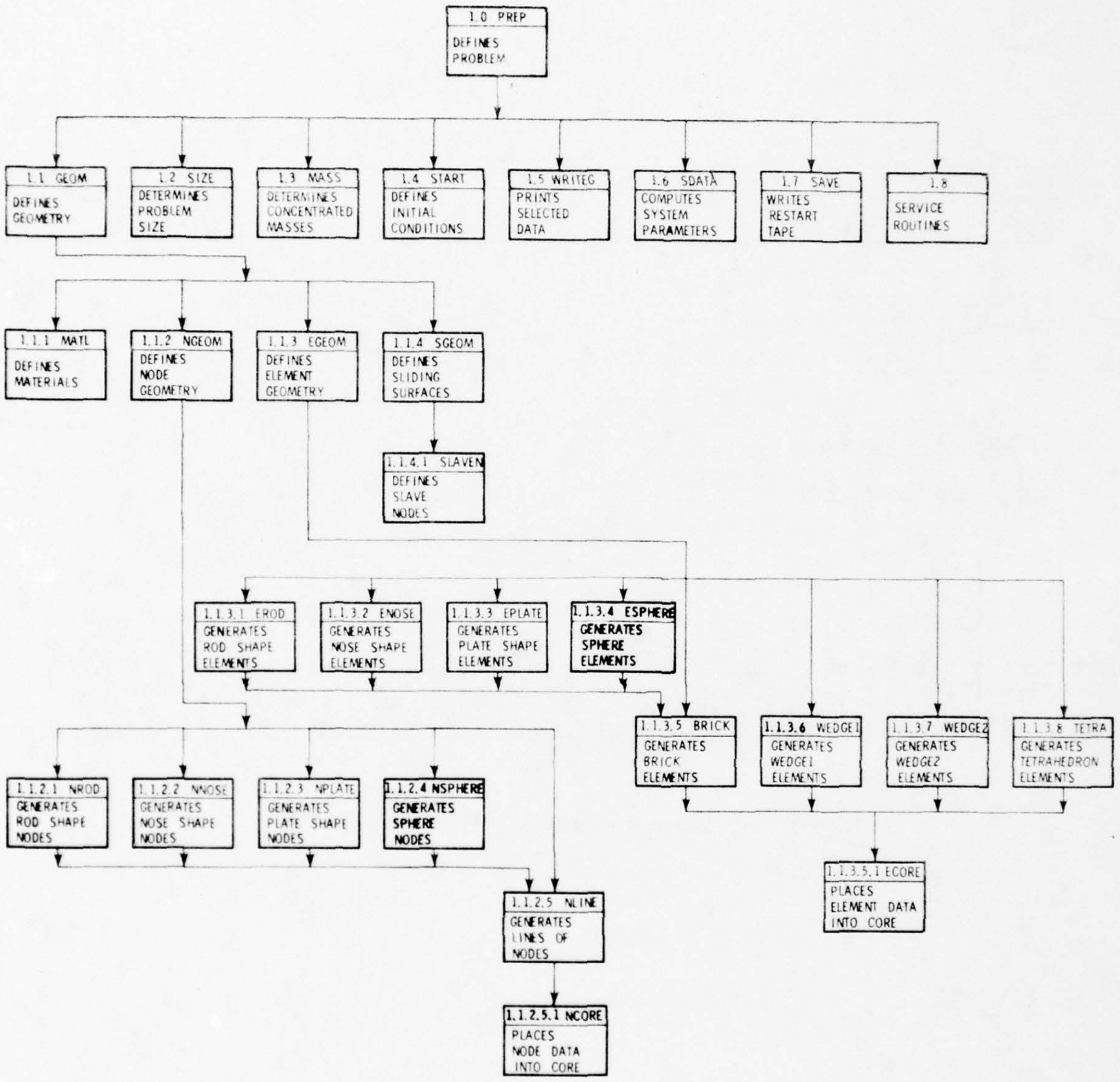


Figure 4. EPIC-3 Interface Diagram



1.0 PREP DEFINES PROBLEM
1.1 GEOM DEFINES GEOMETRY
1.2 SIZE DETERMINES PROBLEM SIZE
1.3 MASS DETERMINES CONCENTRATED MASSES
1.4 START DEFINES INITIAL CONDITIONS
1.5 WRITEG PRINTS SELECTED DATA
1.6 SDATA COMPUTES SYSTEM PARAMETERS
1.7 SAVE WRITES RESTART TAPE
1.8 SERVICE ROUTINES
1.1.1 MATL DEFINES MATERIALS
1.1.2 NGEOM DEFINES NODE GEOMETRY
1.1.3 EGEOM DEFINES ELEMENT GEOMETRY
1.1.4 SGEOM DEFINES SLIDING SURFACES
1.1.4.1 SLAVEN DEFINES SLAVE NODES
1.1.3.1 EROD GENERATES ROD SHAPE ELEMENTS
1.1.3.2 ENOSE GENERATES NOSE SHAPE ELEMENTS
1.1.3.3 EPLATE GENERATES PLATE SHAPE ELEMENTS
1.1.3.4 ESPHERE GENERATES SPHERE ELEMENTS
1.1.3.5 BRICK GENERATES BRICK ELEMENTS
1.1.3.6 WEDGE1 GENERATES WEDGE1 ELEMENTS
1.1.3.7 WEDGE2 GENERATES WEDGE2 ELEMENTS
1.1.3.8 TETRA GENERATES TETRAHEDRON ELEMENTS
1.1.3.5.1 ECORE PLACES ELEMENT DATA INTO CORE
1.1.2.1 NROD GENERATES ROD SHAPE NODES
1.1.2.2 NNOSE GENERATES NOSE SHAPE NODES
1.1.2.3 NPLATE GENERATES PLATE SHAPE NODES
1.1.2.4 NSPHERE GENERATES SPHERE NODES
1.1.2.5 NLINE GENERATES LINES OF NODES
1.1.2.5.1 NCORE PLACES NODE DATA INTO CORE

Figure 5. Preprocessor Hierarchy Chart

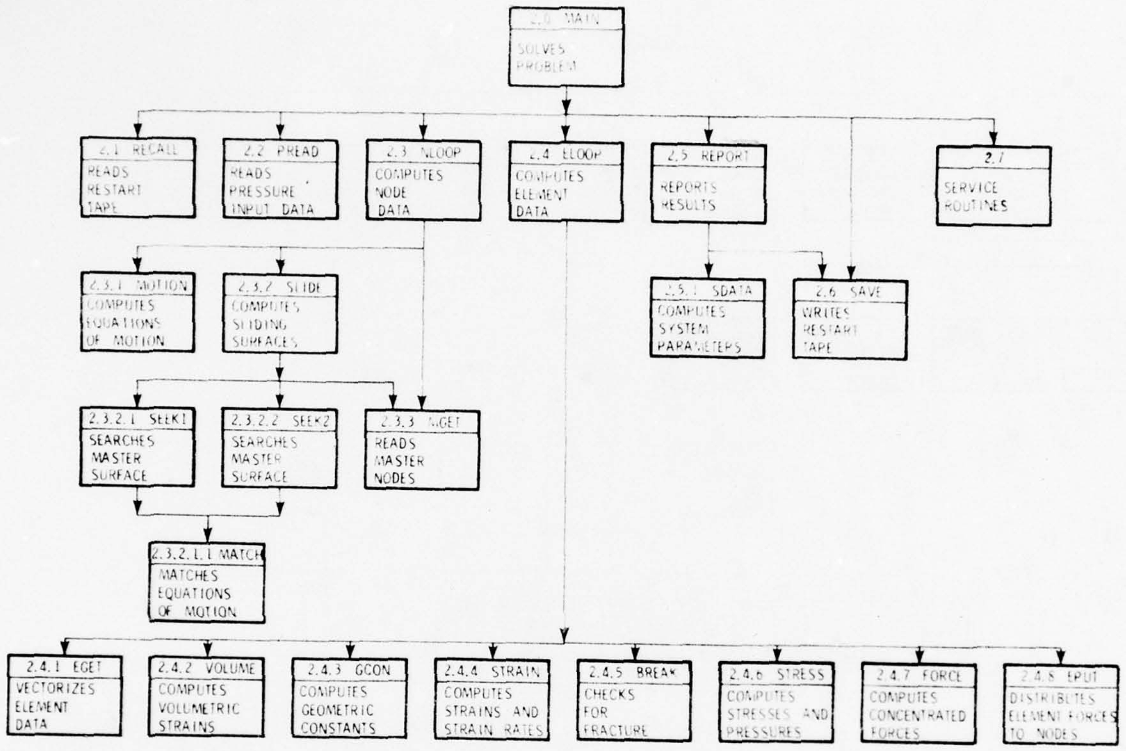


Figure 6. Main Routine Hierarchy Chart

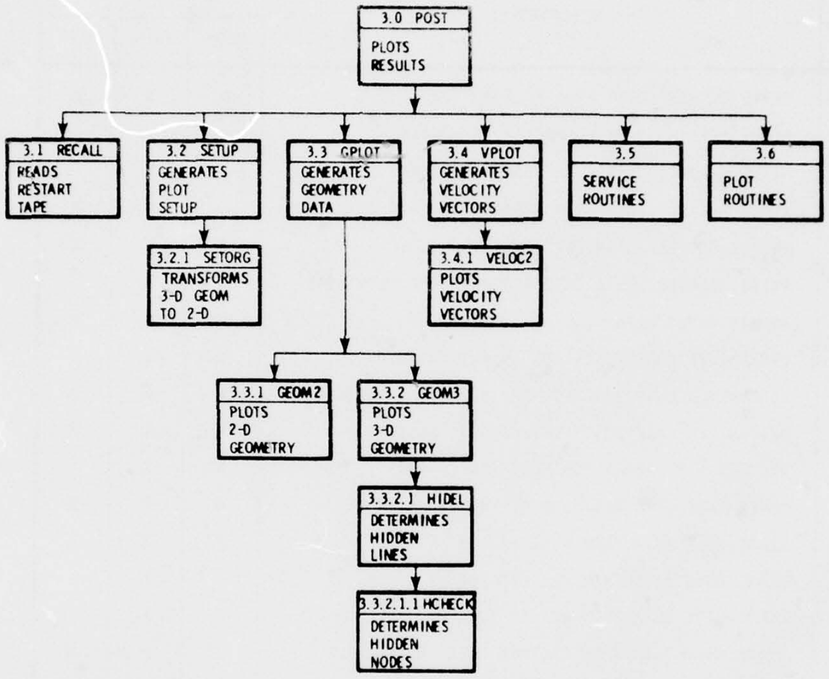


Figure 7. Postprocessor Hierarchy Chart

SERVICE ROUTINE	DESCRIPTION	CONTAINED IN		
		PREP	MAIN	POST
EREAD	READS ELEMENT DATA FROM FILE TO CORE	X	X	X
EREAD1	READS ELEMENT DATA FROM FILE TO BUFFER		X	
EREAD2	READS ELEMENT DATA FROM BUFFER TO CORE		X	
ERTAPE	READS ELEMENT DATA FROM RESTART TAPE TO CORE		X	X
EWRITE	WRITES ELEMENT DATA ONTO FILE FROM CORE	X	X	X
EWTAPE	WRITES ELEMENT DATA ONTO RESTART TAPE FROM CORE	X	X	
HDATA	PRINTS INPUT DATA	X	X	X
LBPOS	DETERMINES CORE POSITIONS OF ELEMENT BLOCK	X	X	X
LPOS	DETERMINES CORE POSITION OF INDIVIDUAL ELEMENT	X	X	X
NBPOS	DETERMINES CORE POSITIONS OF NODE BLOCK	X	X	X
NFIX	DECODES IXYZ ARRAY FOR NODE INDICATORS	X	X	
NPOS	DETERMINES CORE POSITION OF INDIVIDUAL NODE	X	X	X
NREAD	READS NODE DATA FROM FILE TO CORE	X	X	X
NREAD1	READS NODE DATA FROM FILE TO BUFFER		X	
NREAD2	READS NODE DATA FROM BUFFER TO CORE		X	
NRTAPE	READS NODE DATA FROM RESTART TAPE TO CORE		X	X
NWRITE	WRITES NODE DATA ONTO FILE FROM CORE	X	X	X
NWTAPE	WRITES NODE DATA ONTO RESTART TAPE FROM CORE	X	X	
PERSPC	TRANSFORMS 3-D COORDINATES TO 2-D PERSPECTIVE			X
SWITCH	REWINDS AND INTERCHANGES TWO FILES	X	X	X

Figure 8. Service Routines

ROUTINES	INCLUDED IN			LABELED COMMON BLOCKS													SERVICE ROUTINES CALLED																			
	PREP	MAIN	POST	BOOKKP	EBUFR	EBUFW	ELEMNT	FILES	MATERL	MISC	NBUFR	NBUFW	NODE	PLOTD	SLIDED	VECTOR	EREAD	EREAD1	EREAD2	ERTAPE	EWRITE	EWTAPE	HDATA	LBPOS	LPOS	NBPOS	NFIX	NPOS	NREAD	NREAD1	NREAD2	NRTAPE	NWRITE	NWTAPE	PERSPC	SWITCH
BREAK		X					X		X																											
BRICK	X																																			
ECORE	X			X			X														X			X	X											
EGEOM	X			X			X	X													X															X
EGET		X					X						X			X												X								
ELOOP		X		X				X		X			X				X	X	X		X			X		X	X		X	X	X		X			X
ENOSE	X							X																												
EPLATE	X							X																												
EPUT		X					X						X			X												X								
EREAD	X	X	X	X	X		X	X																X												
EREAD1		X		X	X		X	X																X												
EREAD2		X		X	X		X	X																X												
ERTAPE		X	X	X	X		X	X																X												
EROD	X							X																												
ESPHER	X							X																												
EWRITE	X	X	X	X		X	X	X																X												
EWTAPE	X	X		X		X	X	X																X												
FORCE		X					X									X																				
GCON		X					X			X						X																				
GEOM	X			X				X																												
GEOM2			X										X	X																						
GEOM3			X											X																					X	
GPLOT			X	X			X	X		X			X	X			X							X			X	X								
HCHECK			X							X																										
HDATA	X	X	X																																	
HIDEL			X	X				X		X				X																						
LBPOS	X	X	X	X																					X											
LPOS	X	X	X	X																																
MAIN		X		X				X		X													X													
MASS	X			X			X	X	X				X				X				X			X				X	X				X			X
MATCH		X								X					X																					
MATL	X							X	X																											
MGET		X						X					X		X													X								
MOTION		X								X			X																							
NBPOS	X	X	X	X																								X								
NCORE	X			X									X													X		X					X			
NFIX	X	X																																		
NGEOM	X			X				X																									X			X
NLINE	X																																			

Figure 9. EPIC-3 Subroutines

ROUTINES	INCLUDED IN			LABELED COMMON BLOCKS													SERVICE ROUTINES CALLED																			
	PREP	MAIN	POST	BOOKKP	EBUFR	EBUFW	ELEMNT	FILES	MATERL	MISC	NBUFR	NBUFW	NODE	PLOTD	SLIDED	VECTOR	EREAD	EREAD1	EREAD2	ERTAPE	EWRITE	EWTAPE	HDATA	LBPOS	LPOS	NBPOS	NFIX	NPOS	NREAD	NREAD1	NREAD2	NRTAPE	NWRITE	NWTAPE	PERSPC	SWITCH
NLOOP		X		X				X		X					X											X			X	X	X		X			X
NNOSE	X			X				X																												
NPLATE	X			X				X																												
NPOS	X	X	X	X																																
NREAD	X	X	X	X				X			X		X													X										
NREAD1		X		X				X			X		X													X										
NREAD2		X		X				X			X		X													X										
NROD	X			X				X																												
NRTAPE		X	X	X				X			X		X													X										
NSPHER	X			X				X																												
NWRITE	X	X	X	X				X				X	X													X										
NWTAPE	X	X		X				X				X	X													X										
PERSPC			X																																	
PREAD		X		X			X	X		X							X				X			X												X
POST			X	X				X		X				X									X													
PREP	X			X				X		X													X													
RECALL		X	X	X				X	X	X			X							X	X											X	X			X
REPORT		X		X			X	X	X	X			X				X							X		X			X							
SAVE	X	X		X				X	X	X			X				X					X							X					X		
SDATA	X	X		X			X	X		X			X				X							X		X			X							
SEEK1		X											X		X												X									
SEEK2		X						X					X		X												X									
SETORG			X																																	
SETUP			X							X				X																					X	
SGEOM	X			X				X					X																X				X			X
SIZE	X			X			X	X		X			X				X				X			X												X
SLAVEN	X			X				X					X													X	X	X	X				X			
SLIDE		X		X									X		X																					
START	X			X				X		X			X													X	X		X				X			X
STRAIN		X					X									X																				
STRESS		X					X		X	X						X																				
SWITCH	X	X	X																																	
TETRA	X																																			
VELOC2			X										X	X																					X	
VOLUME		X					X			X						X																				
VPLOT			X	X				X		X			X	X												X			X							
WEDGE1	X																																			
WEDGE2	X																																			
WRITEG	X			X			X	X					X				X							X		X			X							

Figure 9. EPIC-3 Subroutines (Concluded)

transportability of a vector-oriented FORTRAN program to other compilers is not endangered. During execution on a vector computer, a vector loop can run many times as fast as its equivalent scalar loop. However, there are certain problems that prevent a loop from vectorizing.

To be vectorizable, a computational loop should manipulate arrays and store the results of computations in arrays. Loops that contain branches such as GO TO's and IF's, cannot currently be vectorized. The use of temporary variables, CALL statements or pointers (an array position identifier obtained from another array) also prohibits vectorization.

The following general rules apply for vector loops:

- Keep subscripts simple and explicit.
- Do not use temporary variables, pointers, IF's, GO TO's, or CALL's.
- Use mathematical functions for which there are vectorized versions in the target vector computer.
- If a large loop contains only a few unvectorizable statements, try to rewrite it as two or more loops, with at least one being vectorizable.

Much of the EPIC-3 code is currently vectorized based on the application of the aforementioned rules. Since most of the element computations depend on nodal data such as positions and velocities, the nodal data are redefined in terms of element variables such that pointers are not required for the primary-element computational loops. The gathering of nodal data into element arrays is performed in subroutine EGET, and the subsequent distribution of element data back to the nodal arrays (concentrated forces) is performed in subroutine EPUT.

4. CODING GUIDELINES

Most of the EPIC-3 program (with the exception of the geometry generators) is written using structured programming guidelines. Other exceptions occur in some of the primary computational loops where computational efficiency or vectorization would be impaired if the structured programming guidelines were incorporated. Every attempt has been made to provide consistency and clarity in all the subroutines. An example of the coding is provided with subroutine MOTION, shown in Figure 10.

5. DATA STRUCTURE

The significant data within the EPIC-3 software are organized in labeled common blocks. These blocks provide the basic means for communication between subroutines. The block labels, in addition to the variable names themselves, are intended to be as self-explanatory as possible. Figure 9 lists the common block usage by subroutine. The common blocks and their contents are:

/BOOKKP/	-	All bookkeeping information such as block sizes, number of blocks, total number of elements, and total number of nodes.
/EBUFR/	-	Buffer for element data that are being read.
/EBUFW/	-	Buffer for element data that will subsequently be written onto an element file.
/ELEMNT/	-	Data arrays for one block of elements.
/FILES/	-	Variables used to store the logical unit designations for all EPIC-3 files.

```
      SUPROUTINE MCTION (IFIRST,ILAST)
C
C LATEST REVISION MARCH 31, 1978
C
C THIS SUBROUTINE CCMPUTES EQUATIONS OF MCTION (VELOCITIES AND
C   POSITIONS) FOR NODES IN CORE POSITICNS IFIRST TO ILAST
C
      COMMON /MISC/     NCYCLE,TIME,DT,CTBAR,DTNEXT,CTLAST,ENERGY,ETEST,
     1                  EPSLON,NPRES,IPRES,IBANDF,INCORE,PFACT(50),
     2                  PFTIME(50)
      COMMON /NODE/     X(1600),Y(1600),Z(1600),XDOT(1600),
     1                  YDOT(1600),ZDOT(1600),PX(1600),PY(1600),
     2                  PZ(1600),AMASS(1600),PMASS(1600),IXYZ(1600),
     3                  MASTER(1600)
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C DO FOR ALL NODES (VECTOR LOOP)
C   COMPUTE UPDATED VELOCITIES
C   COMPUTE UPDATED POSITIONS
C   ZERO FORCES FOR NEXT CYCLE
C END DO
C
      DO 10 I=IFIRST,ILAST
        XDOT(I) = XDOT(I) + (PX(I)/AMASS(I))*DTBAR
        YDOT(I) = YDOT(I) + (PY(I)/AMASS(I))*DTBAR
        ZDOT(I) = ZDOT(I) + (PZ(I)/AMASS(I))*DTBAR
        X(I)    = X(I) + XDOT(I)*DT
        Y(I)    = Y(I) + YDOT(I)*DT
        Z(I)    = Z(I) + ZDOT(I)*DT
        PX(I)   = 0.
        PY(I)   = 0.
        PZ(I)   = 0.
 10     CONTINUE
C     END DO
      RETURN
      END
```

Figure 10. Example of Structured Programming Coding Format

/MATERL/	-	Material properties.
/MISC/	-	Miscellaneous data needed by multiple routines.
/NBUFR/	-	Buffer for node block data that are being read.
/NBUFW/	-	Buffer for node block data that will subsequently be written onto a node file.
/NODE/	-	Data arrays for the entire bandwidth of node blocks.
/PLOTD/	-	Assorted plot data.
/SLIDED/	-	Sliding surface data.
/VECTOR/	-	Arrays needed for various element vector loops.

Files are used extensively by the EPIC-3 software. The node, element, and sliding surface files are designed to minimize the program's core storage requirements by providing for the storage and subsequent retrieval of data that are not currently needed for calculations. Typically, data will be read from files into common block locations, it will be processed, and then the updated data will be written to other files. A summary of the various files is given in Section IV.

6. BANDWIDTH DETERMINATION

Since three-dimensional problems often require many nodes and elements, it is not generally possible to contain all data within the central memory core of existing computers. As a result, it is necessary to use disk files for intermediate storage. Since there is interaction between various nodes for sliding surface computations, and since element computations require various node data, it is necessary to provide a logical, orderly technique whereby the appropriate element and node data are in core at the proper time during the computations.

For each cycle of numerical integration, there are three computational loops. A brief discussion of each follows:

- The first loop computes the equations of motion of the nodes (Subroutine MOTION). It does not include the sliding surface computations. The computations for each node are self contained; they do not depend on data from other nodes or elements. Therefore, this computation has no effect on the data handling or central memory requirements.

- The second loop adjusts the equations of motion for the sliding surface nodes (Subroutine SLIDE). The computations for each slave node do not depend on any element data; they do, however, interact with data from other nodes (the master nodes). Therefore, for a specific sliding surface, it is necessary that all master nodes be in core when each slave node is processed.

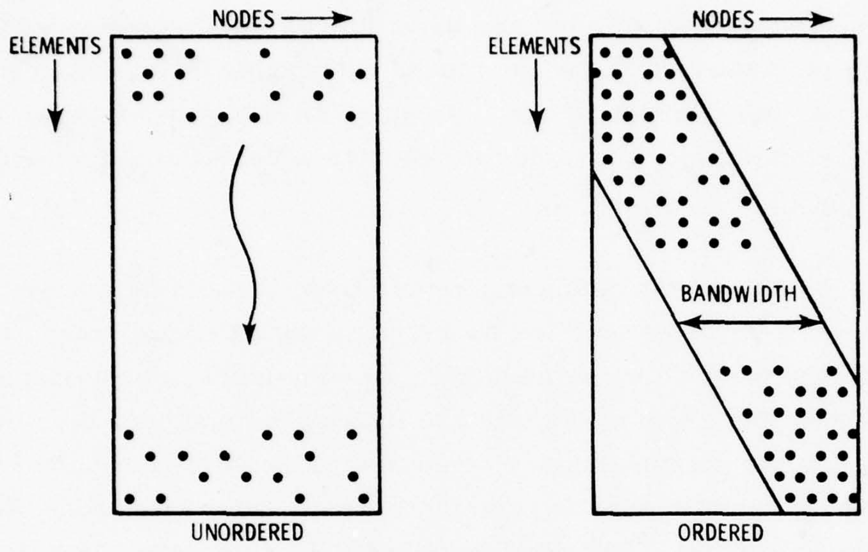
- The third loop computes various data for the elements (Subroutine ELOOP). The computations for each element do not depend on data from any other element. They do, however, depend on node data. The computations for a specific element require the position and velocity data for the four nodes which define that element. The element then uses this data (together with material property data) to generate concentrated forces which it distributes to the four nodes. Therefore, for a specific element being processed, it is necessary for the four nodes of an element to be in core as that element is being processed.

The second and third loops both place requirements on the data handling and central memory requirements. It is from these requirements that the concept of a bandwidth is introduced. The third loop is the most complicated since both node and element data must be in central memory at the proper time. The discussion which follows is for this third (element) loop.

An unordered arrangement of nodes and elements is shown at upper left in Figure 11. Each element along the vertical axis is affected by four nodes along the horizontal axis. If the primary computational activity is based on processing element data in a sequential manner, the unordered arrangement does not restrict the location of the appropriate four nodes for that element. As a result, if the required node data are not in core but must be brought in from disk storage in a random manner, the computations are inefficiently performed.

In contrast, an ordered arrangement of nodes and elements is shown at upper right in Figure 11. Here it can be seen that the four nodes for each element are restricted to fall within a range defined as the bandwidth. This means that, as each element is processed, it is necessary to provide access to only a limited number of nodes contained within the bandwidth. This ordered arrangement is obtained by numbering the nodes and elements in a logical and consistent manner. Generally, the bandwidth for the element loop is minimized if the nodes are increasingly numbered in a direction normal to the cross section with the smallest number of nodes.

The lower portion of Figure 11 shows an ordered arrangement of nodes and elements where the data are represented by N blocks of node data and M blocks of element data. These blocks contain data for groups of nodes and elements. This is done to simplify the transfer of data between disk files and central memory and also to allow the vectorized computational loops to act on an entire block of data. The bandwidth for a particular element block is the number of consecutive node blocks needed to contain both the lowest



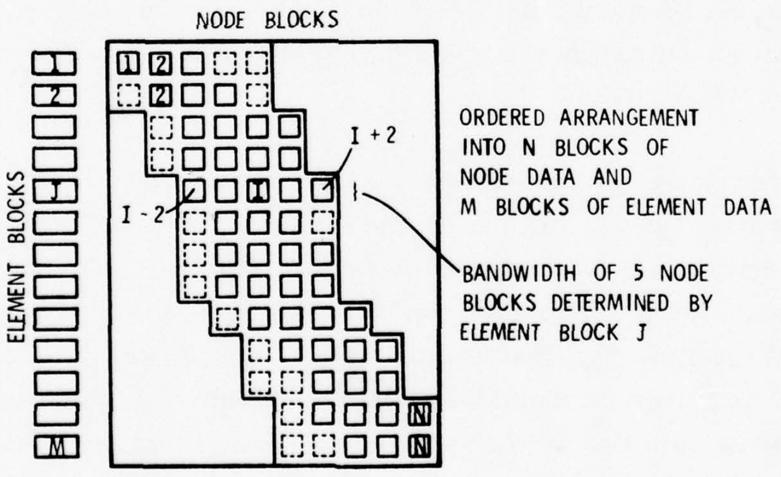

NODE BLOCKS
ELEMENT BLOCKS
1
2
J
M
1
2
2
I
N
N
I + 2
I - 2
ORDERED ARRANGEMENT INTO N BLOCKS OF NODE DATA AND M BLOCKS OF ELEMENT DATA
BANDWIDTH OF 5 NODE BLOCKS DETERMINED BY ELEMENT BLOCK J

Figure 11. Bandwidth Representation for Node-Element Connectivity

node number and the highest node number used by any element in that element block or any preceding block. A bandwidth of five node blocks is shown to contain the required node data for each element block in Figure 11. The node blocks represented with solid lines are those blocks which are needed by the corresponding element block, and the blocks represented with dashed lines are not needed by the element blocks. It can be seen that this bandwidth is determined by element block J, since it requires access to the five node blocks from block I-2 to block I+2.

Another requirement for the bandwidth comes from the sliding surface computations. Since this involves only the nodes, it is not as complicated as the previously defined bandwidth obtained from the elements. For a specified sliding surface, if the slave nodes all have lower node numbers than the associated nodes of the master surface, then the bandwidth is the number of consecutive node blocks needed to contain all slave and master nodes for that particular sliding surface. The final required bandwidth is the maximum determined from the elements and the sliding surfaces. Since it is desirable to minimize this bandwidth, the geometry generators in the Preprocessor are developed accordingly.

The current version of the code contains 64 nodes per node block and 64 elements per element block. (The node and element block sizes do not have to be identical, however.) This size is well suited for vector processing on the CRAY computer. It also provides for the transfer of a significant amount of data such that the time required to initiate the Input/Output activity is relatively small. It should be noted that some problems will have few enough nodes so all node data can be core-contained. If adequate core is provided, the transfer of nodal data between files and core will be eliminated. The savings in the associated Input/Output and Central Processor time for the core-contained option must be compared to the advantages associated with using a smaller core size which contains only a bandwidth of nodal data.

7. COMPUTATIONAL SEQUENCE

The primary computational sequences for EPIC-3 have been designed to minimize problem size requirements and to enhance vector processing. A schematic representation of the data manipulation is shown in Figure 12. The central memory core contains one block of element data and a bandwidth of node blocks. Node and element buffers for read and write activity are required, and storage space is provided for the material and sliding surface data.

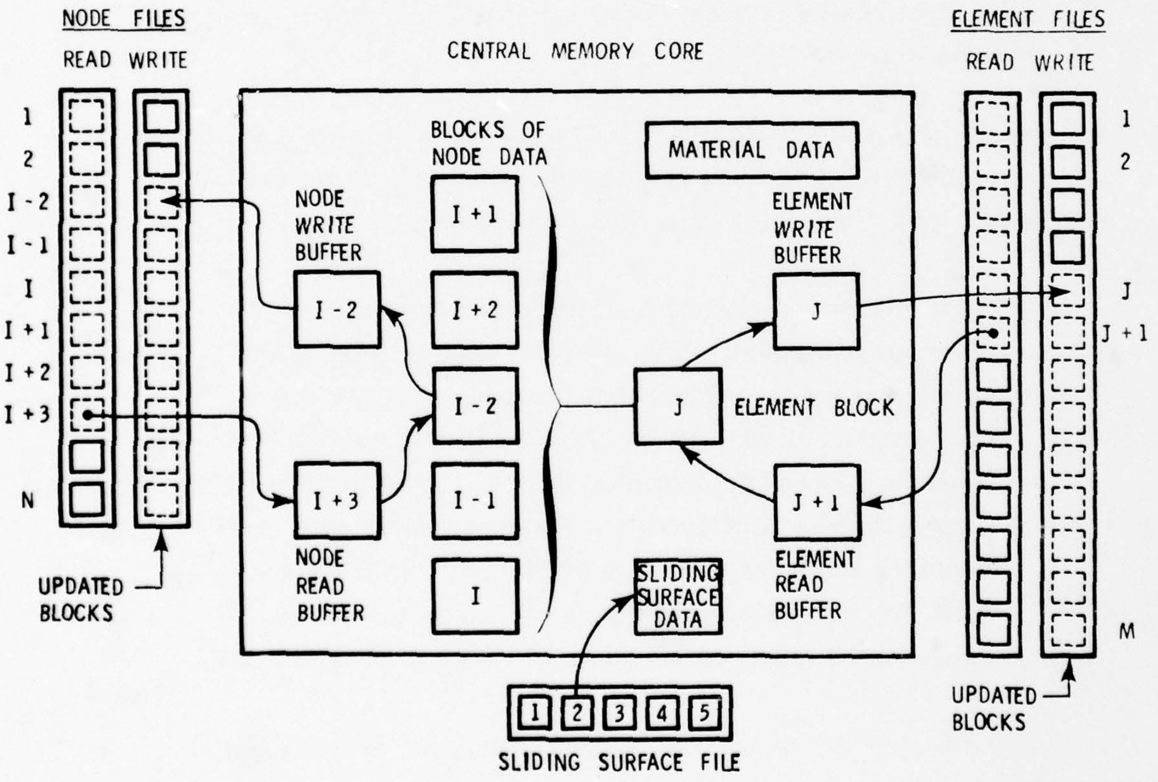


Figure 12. Program Logic and Memory Requirements

The following process describes the important steps which occur during an element loop (Subroutine ELOOP). These steps are illustrated in Figures 11 and 12 where the bandwidth is five node blocks.

- Initial Step to Set Up Processing - A complete bandwidth of node blocks is read into core from the current node read file. An additional node block, if it exists, will be read into the node read buffer. The first element block must also be read into core from the element read file and the buffered read of the next one is initiated. In Figures 11 and 12, this would mean node blocks 1 to 5 are read into core, and node block 6 is read into the node read buffer. Likewise, element block 1 is read into core, and element block 2 is read into the element read buffer.

- Basic Processing Sequence (Done for Each Element Block) - The following must be repeatedly done until the maximum node block needed for this element block is in core:

  The lowest numbered node block in core is transferred to the node write buffer and written on the node write file. Since it is no longer within the bandwidth, it will not be affected by subsequent elements. Then the next node block is transferred from the node read buffer to the vacated core space and another buffered node read is begun. In Figures 11 and 12, before element block J can be processed, node block I-3 must be replaced with node block I+2.

  The element computations for this block can now be performed since the required node data are in core. These computations are summarized under Subroutine ELOOP in Figure 6.

The updated element block is written on the element write file and the next element block is transferred from the element read buffer to core. Then the next buffered element read is begun. In Figures 11 and 12, element block J is replaced by element block J+1.

- Final Step to Complete the Processing - Any remaining node blocks in core must be written on the node write file. This would be node blocks N-4 to N as shown in Figures 11 and 12.

  The logical unit designators for the node read and write files are interchanged so that subsequent processing will read from the updated file. Likewise, the element files are interchanged.

The two node loops (Subroutines MOTION and SLIDE) are less complex since there is no interaction with element data. For the MOTION loop the node blocks are read from one node file, the computations are performed, and the updated results are written onto the other node file.

For the sliding surface computations, a bandwidth of node blocks must always be in core. If the computations are initiated with the slave nodes in the lowest node block, then the affected higher-numbered master nodes must always be in core. When the last slave node in a specific sliding surface has been processed, the data (master node numbers, etc.) for the next sliding surface are read into core from the sliding file. This requires the slave nodes of a subsequent sliding surface to be numbered higher than those of a preceding sliding surface.

8. THREE-DIMENSIONAL PLOTTING ROUTINES

This subsection summarizes the techniques used to obtain the three-dimensional plots. For each three-dimensional plot request, one call is made to the SETORG subroutine which develops the transformation from 3-space to a 2-space plotting plane. Since the user specifies an eye point and an origin (the point looked at), SETORG can compute the equation for the plotting or viewing plane which is perpendicular to the line of sight and contains the origin. Subsequent calls to the PERSPC subroutine provide 2-space coordinates for any point as it is projected onto the viewing plane.

The x, y, and z maximum and minimum values are used to scale the 3-D plot. The eight combinations of coordinates define the corner nodes of a box in space. PERSPC is called to project this box onto the viewing plane using the transformation defined for this particular 3-D plot request. The vertical dimension of the projected figure will be scaled to fit within 10 inches. The horizontal dimension will use the same scale to avoid distortion; the length of the plot is unlimited.

Plotting requires one complete pass through the element file. All four triangular faces of each element are examined. If a triangle fits within the maximum and minimum extremes, a corresponding data record is written onto a new triangle file. The three nodes are listed in ascending order, followed by their x, y, z coordinates. The triangle file is sorted on ascending node numbers and then a new file containing only exterior triangles is written. The interior triangles are easily identified and eliminated by the presence of two or more consecutive, identical records on the sorted triangle file.

The exterior triangle file is further processed if the hidden line option is requested. One complete node loop is made to flag all exterior nodes (those contained in the external triangles). To ascertain an exterior node's visibility, each exterior triangle must be examined to see whether its 2-space projection contains the node and if so, whether the node is behind the triangle. Once all visible nodes are determined, one complete triangle pass is made to write those triangles having three visible nodes and to create degenerate (two-node) triangles when two of the three nodes are visible. With this method, any line of a triangle is eligible for eventual plotting if its endpoints are visible. Unfortunately, some erroneous lines can occur at the corners of a plot if the endpoints of a line are visible, but the line itself is hidden by another external triangle.

The final triangle file, with or without hidden lines, is now processed for plotting. For each triangle, the PERSPC subroutine is called three times to transform the three nodes to their 2-space representation on the plotting plane. Then standard plot subroutines are called to draw lines between the three nodes.

SECTION IV
PROGRAM USER INSTRUCTIONS

This section provides instructions for the user. Input data for the Preprocessor, the Main Routine, and the Postprocessor are given in the first three subsections. Then the output data are described and instructions for changing the dimensions of the program, the file designations, and central processor time estimates are given.

1. INPUT DATA FOR THE PREPROCESSOR

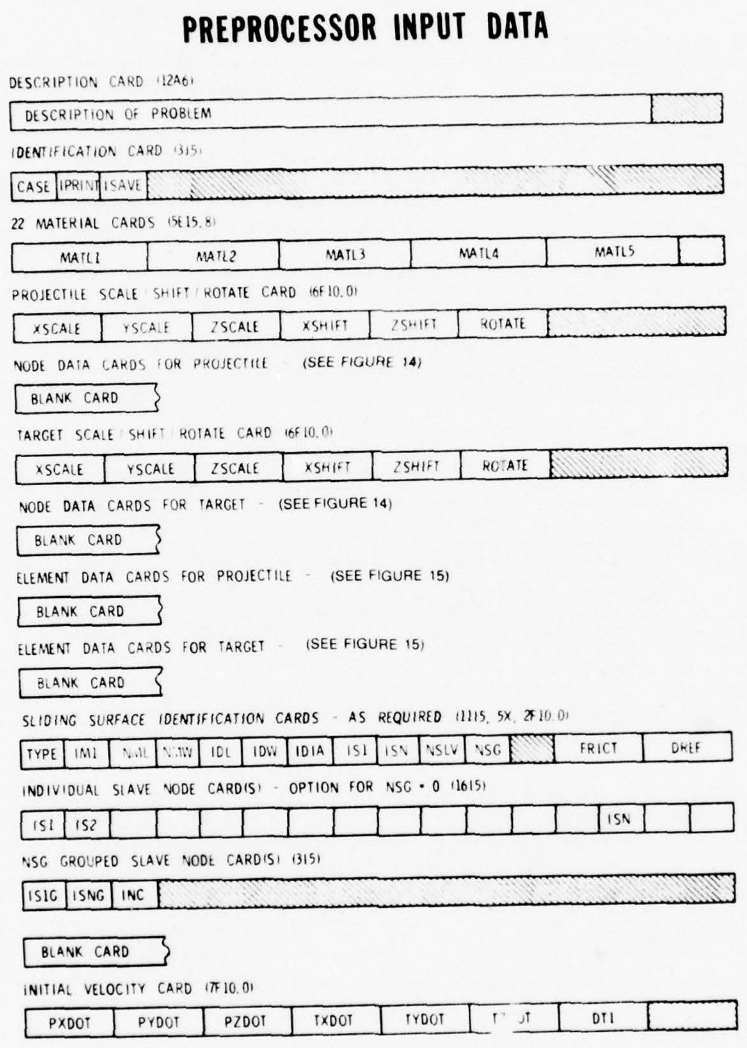
The functions of the Preprocessor are to define the initial geometry and velocity conditions and to determine the memory size requirements for the Main Routine. A summary of the input data for the Preprocessor is given in Figure 13, and details of the node and element input data are given in Figures 14 and 15. These Figures correspond to paragraphs a, b, and c below. The descriptions which follow are for the data in Figure 13. Consistent units must be used.

a. Preprocessor Input Data Summary

- Description Card (12A6) - A description of the problem provided by the user.

- Identification Card (315) -

CASE	=	Case number for run identification
I PRINT	=	0 will not print individual data for each node and element.
		1 will print individual data
I SAVE	=	0 will not write results on restart tape.
		1 will write on restart tape.



PREPROCESSOR INPUT DATA

DESCRIPTION CARD (12A6)

DESCRIPTION OF PROBLEM	▨

IDENTIFICATION CARD (3I5)

CASE	IPRINT	ISAVE	▨

22 MATERIAL CARDS (5E15.8)

MATL1	MATL2	MATL3	MATL4	MATL5	

PROJECTILE SCALE / SHIFT / ROTATE CARD (6F10.0)

XSCALE	YSCALE	ZSCALE	XSHIFT	ZSHIFT	ROTATE	▨

NODE DATA CARDS FOR PROJECTILE - (SEE FIGURE 14)

BLANK CARD

TARGET SCALE / SHIFT / ROTATE CARD (6F10.0)

XSCALE	YSCALE	ZSCALE	XSHIFT	ZSHIFT	ROTATE	▨

NODE DATA CARDS FOR TARGET - (SEE FIGURE 14)

BLANK CARD

ELEMENT DATA CARDS FOR PROJECTILE - (SEE FIGURE 15)

BLANK CARD

ELEMENT DATA CARDS FOR TARGET - (SEE FIGURE 15)

BLANK CARD

SLIDING SURFACE IDENTIFICATION CARDS - AS REQUIRED (11I5, 5X, 2F10.0)

TYPE	IM1	NML	NMW	IDL	IDW	IDIA	IS1	ISN	NSLV	NSG	▨	FRICT	DREF

INDIVIDUAL SLAVE NODE CARD(S) - OPTION FOR NSG = 0 (16I5)

IS1	IS2												ISN		

NSG GROUPED SLAVE NODE CARD(S) (3I5)

ISIG	ISNG	INC	▨

BLANK CARD

INITIAL VELOCITY CARD (7F10.0)

PXDOT	PYDOT	PZDOT	TXDOT	TYDOT	TZDOT	DT1	

Figure 13. Preprocessor Input Data

NODE INPUT DATA

TYPE OF NODES DESCRIPTION CARD (2I5, 6F10.0, F7.0, 3I1)

1	NNODE	X1	Y1	Z1	XN	YN	ZN	EXPAND			

IX, IY, IZ

ROD NODE IDENTIFICATION CARD (I5)

2	▨

ROD NODE DESCRIPTION CARD (4I5, 3F10.0)

NOR	NIR	NPLN	IRAD	ZTOP	ZBOT	EXPAND	▨

ROD NODE RADII CARD FOR IRAD = 0 (4F10.0)

ROTOP	RITOP	ROBOT	RIBOT	▨

ROD NODE TOP RADII CARD(S) FOR IRAD = 1 (8F10.0)

RT(NIR)						RT(NOR)	

ROD NODE BOTTOM RADII CARD(S) FOR IRAD = 1 (8F10.0)

RB(NIR)						RB(NOR)	

NOTE: IF NIR = 0 BEGIN TOP AND BOTTOM RADII CARDS WITH RT(1) AND RB(1)

NOSE NODE IDENTIFICATION CARD (I5)

3	▨

NOSE NODE DESCRIPTION CARD (4I5, 4F10.0)

NOR	NIR	INOSE	IRAD	ROTOP	RITOP	ZTOP	ZMIN	▨

NOSE NODE TOP RADII CARD(S) FOR IRAD = 1 (8F10.0)

RT(NIR)						RT(NOR)	

NOSE NODE ZMIN CARD(S) FOR IRAD = 1 (8F10.0)

ZM(NIR)						ZM(NOR)	

NOTE: IF NIR = 0 BEGIN TOP RADII AND ZMIN CARDS WITH RT(1) AND ZM(1)

FLAT PLATE NODE IDENTIFICATION CARD (I5)

4	▨

FLAT PLATE NODE DESCRIPTION CARD (6I5, 5F10.0)

NX	NY	NZ	NXEND	NYEND	IY	X-EXPAND	X-PART	Y-EXPAND	Y-PART	Z-EXPAND

FLAT PLATE NODE SIZE CARD (6F10.0)

X1	Y1	Z1	XN	YN	ZN	▨

SPHERE NODE IDENTIFICATION CARD (I5)

5	▨

SPHERE NODE DESCRIPTION CARD (2I5, 5X, I5, 3F10.0)

NOR	NIR	▨	IRAD	RO	RI	ZCG	▨

SPHERE NODE RADII CARD(S) FOR IRAD = 1 (8F10.0)

R(NIR)					R(NOR)		

NOTE: IF NIR = 0 BEGIN RADII CARD WITH R(1)

Figure 14. Node Input Data

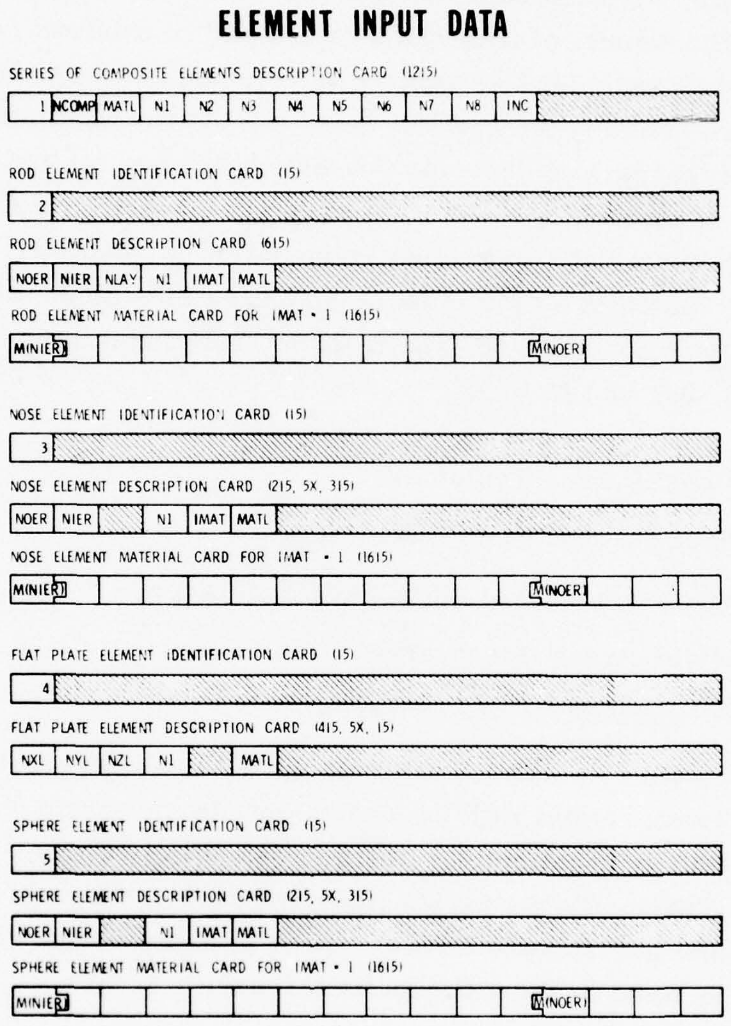

ELEMENT INPUT DATA
SERIES OF COMPOSITE ELEMENTS DESCRIPTION CARD (12I5)
1 NCOMP MATL N1 N2 N3 N4 N5 N6 N7 N8 INC
ROD ELEMENT IDENTIFICATION CARD (I5)
2
ROD ELEMENT DESCRIPTION CARD (6I5)
NOER NIER NLAY N1 IMAT MATL
ROD ELEMENT MATERIAL CARD FOR IMAT = 1 (16I5)
M(NIER) M(NOER)
NOSE ELEMENT IDENTIFICATION CARD (I5)
3
NOSE ELEMENT DESCRIPTION CARD (2I5, 5X, 3I5)
NOER NIER N1 IMAT MATL
NOSE ELEMENT MATERIAL CARD FOR IMAT = 1 (16I5)
M(NIER) M(NOER)
FLAT PLATE ELEMENT IDENTIFICATION CARD (I5)
4
FLAT PLATE ELEMENT DESCRIPTION CARD (4I5, 5X, I5)
NXL NYL NZL N1 MATL
SPHERE ELEMENT IDENTIFICATION CARD (I5)
5
SPHERE ELEMENT DESCRIPTION CARD (2I5, 5X, 3I5)
NOER NIER N1 IMAT MATL
SPHERE ELEMENT MATERIAL CARD FOR IMAT = 1 (16I5)
M(NIER) M(NOER)

Figure 15. Element Input Data

- 22 Material Cards (5E15. 8) - Each of the 22 material cards provides a specific material characteristic for five materials. For instance, the first card describes the density of the materials. The density of material 1 is entered in columns 1 to 15, the density of material 2 is entered in columns 16 to 30, etc. The second card contains the specific heats for the five materials, etc. It is only necessary to describe the materials used for the analysis. If materials 1 and 3 are used for a specific analysis, all the material 1 data must be specified in columns 1 to 15 and all the material 3 data must be specified in columns 31 to 45. Columns 16 to 30, 46 to 60, and 61 to 75, representing materials 2, 4 and 5, can be left blank.

Card 1	Density (mass/volume)
Card 2	Specific heat (work/mass/degree)
Card 3	Shear modulus of elasticity (force/area)
Card 4	Yield stress (force/area)
Card 5	Ultimate stress (force/area)
Card 6	Strain at which the ultimate stress is achieved. Should be consistent with the equivalent plastic strain definition in Equation (39) which is essentially a true strain. Stress varies linearly between the yield stress and the ultimate stress. The stress for larger strains is equal to the ultimate until fracture occurs.
Card 7	Maximum negative hydrostatic pressure (force/area)
Card 8	Strain rate effect constant, C_1, in Equation (40)
Card 9	Pressure effect constant, C_2, in Equation (40)
Card 10	Pressure effect constant, C_3, in Equation (40)
Card 11	Temperature effect constant, C_4, in Equation (40)

Card 12 Temperature effect constant, C_5, in Equation (40)

Card 13 Hydrostatic pressure constant, K_1, in Equation (42) (force/area)

Card 14 Hydrostatic pressure constant, K_2, in Equation (42) (force/area)

Card 15 Hydrostatic pressure constant, K_3, in Equation (42) (force/area)

Card 16 Gruneisen coefficient, Γ, in Equation (42)

Card 17 Linear artificial viscosity coefficient, C_L, in Equation (45)

Card 18 Quadratic artificial viscosity coefficient, C_Q, in Equation (45)

Card 19	Equivalent strain	If either is exceeded, there is a shear and tensile failure. Positive hydrostatic pressure and viscosity stress capability remain. If Equivalent strain is negative, material behaves like a liquid.
Card 20	Volumetric strain	
Card 21	Equivalent strain	If exceeded, the element fails totally and produces no stresses or pressures.

Card 22 Initial temperature (degrees)

- <u>Projectile Scale/Shift/Rotate Card (6F10.0) -</u>

XSCALE	=	Factor by which the x coordinates of all projectile nodes are multiplied. Applied after the coordinate shifts (XSHIFT, ZSHIFT) described later.

YSCALE	=	Factor by which the y coordinates are multiplied.
ZSCALE	=	Factor by which the z coordinates are multiplied.
XSHIFT	=	Increment added to the x coordinates of all projectile nodes (length). Applied before the scale factors (XSCALE, YSCALE, ZSCALE).
ZSHIFT	=	Increment added to the z coordinates (length).
ROTATE	=	Rotation about the axis (at x = z = 0) of all projectile nodes (degrees). Applied after the coordinate shifts (XSHIFT, ZSHIFT) and the scale factors (XSCALE, YSCALE, ZSCALE).

- Node Data Cards for Projectile - These are described in Section IV-1-b. End with a blank card.

- Target Scale/Shift/Rotate Card (6F10.0) - Same as Projectile Scale/Shift/Rotate Card except it applies to the target nodes.

- Node Data Cards for Target - These are described in Section IV-1-b. End with a blank card.

- Element Data Cards for Projectile - These are described in Section IV-1-c. End with a blank card.

- Element Data Cards for Target - These are described in Section IV-1-c. End with a blank card.

- Sliding Surface Identification Card (11I5, 5X, 2F10.0) - Each sliding surface contains one Identification Card and cards (if required) describing the slave nodes. For each sliding surface all the slave nodes must have lower node numbers than all the master nodes of that particular sliding surface. The slave nodes of a subsequent sliding surface must all have higher node numbers than those of the slave nodes of the previous sliding surface. If there is more than one sliding surface, all data for the first surface are input before beginning data for the second surface. The mass and spacing of the slave nodes should not be significantly greater than that of the master nodes in the initial or deformed geometry. Also, the slave nodes cannot be restrained in the z direction End with a blank card after the last sliding surface

TYPE	=	1 gives the specialized search routine as described in Section II. The slave surface must be above the master surface relative to the z axis, and it must be a single-valued function of x and y.
		2 gives the generalized search routine as described in Section II.
IM1	=	The node number of the first master node which corresponds to node 1 in Figure 3.
NML	=	Number of nodes per row of master nodes. NML is equal to N in Figure 3. Each row of master nodes must have the same number of nodes.
NMW	=	Number of rows of master nodes. NMW is equal to M in Figure 3.
		Note: When looking from the slave node side to the master node side of the sliding surface, each successive row of master nodes is to the left of the preceding row of nodes when viewed from node 1 to node N.

IDL	=	The node number increment along the rows of master nodes. If IM1 = 100, NML = 6, and IDL = 2, then the first row of nodes in the master surface consists of nodes 100, 102, 104, 106, 108, 110.
IDW	=	The node number increment between the first node of each row. If IDW = 20 and IM1, NML and IDL are as described in the preceding description of IDL, then the second row of master nodes consists of nodes 120, 122, 126, 128, 130.
		1 is for the diagonal orientation shown in Figure 3.
IDIA	=	2 is for the other diagonal orientation where the diagonals go in the general direction from the first master node to the last master node.
IS1	=	The first (lowest node number) slave node.
ISN	=	The last (highest node number) slave node. Must have a lower node number than all the master nodes.
NSLV	=	The total number of slave nodes in the sliding surface. If all nodes between IS1 and ISN are slave nodes (NSLV = ISN - IS1 + 1), no additional input data are necessary.
		Otherwise, slave nodes are read individually or in groups.
NSG	=	Numbers of groups of slave nodes to be read. If NSG = 0, the nodes are either consecutive between IS1 and ISN, or they are read individually.

FRICT	=	The coefficient of sliding friction.
DREF	=	A reference distance used to determine the zone of acceptability, as shown in Figure 3. Used only for the generalized search routine (TYPE = 2). This distance should be large enough such that a sphere with a radius of DREF will completely contain each deformed triangular master plane if the center of the sphere coincides with the center of the triangle. Any slave nodes outside this spherical zone will not be considered for that specific triangular plane.

- Individual Slave Node Cards (16I5) - Individual slave nodes are read if the slave nodes are not consecutive between IS1 and ISN, and if the slave nodes are not read in groups (NSG = 0) Slave nodes must be read in ascending order from IS1 to ISN.

- NSG Grouped Slave Node Cards (3I5) -

ISG1	=	The first (lowest number) node in a group of nodes
ISGN	=	The last (highest number) node in a group of nodes
INC	=	The increment between the slave nodes. If ISG1 = 100, ISGN = 5, then nodes 100, 105, 110, 115, 120 will be designated as slave nodes.

- Initial Velocity Card (7F10.0) - This card describes the initial velocity conditions of the projectile and the target. If there are interface nodes which include mass from both the projectile and the target, the velocities of these nodes are adjusted by the program to conserve momenta.

PXDOT	=	Projectile velocity in the x direction (distance/time)

PYDOT	=	Projectile velocity in the y direction
PZDOT	=	Projectile velocity in the z direction
TXDOT	=	Target velocity in the x direction
TYDOT	=	Target velocity in the y direction
TZDOT	=	Target velocity in the z direction
DT1	=	Integration time increment (Δt in equations of motion in Section II) for the first cycle. This must be less than the time required to travel across the minimum altitude of each tetrahedron element at the sound speed of the material in that element.

b. Input Data for Node Geometry

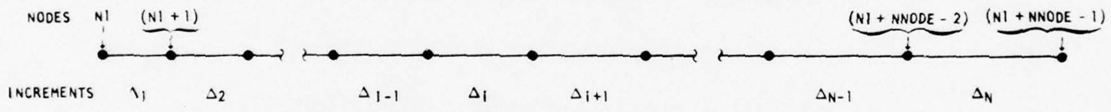
A summary of input data for nodal geometry is given in Figure 14. Nodes may be input as a line of nodes, and/or special shapes consisting of rods, various node geometries, flat plates and spheres. There is no limit to the number of shapes included for the projectile or the target. In all cases the nodes are numbered consecutively. If a node is at the interface of the projectile and the target and contains mass from both the projectile and the target, it must be included with the projectile nodes.

Line of Nodes Card -- Node cards for each line of nodes are supplied as needed by the user. Refer to Figure 16 for the spacing of the nodes.

Line of Nodes Description Card (2I5, 6F10.0, F7.0, 3I1) -

1	=	Identification number for a line of nodes.
NNODE	=	Number of nodes in the row of nodes. The nodes are numbered consecutively.

NUMBER OF INCREMENTS N	EXPAND																	
	.7		.8		.9		1.0		1.1		1.2		1.3		1.4		1.5	
	$\frac{\Delta_1}{\bar{\Delta}}$	$\frac{\Delta_N}{\bar{\Delta}}$	$\frac{\Delta_1}{\Delta}$	$\frac{\Delta_N}{\bar{\Delta}}$	$\frac{\Delta_1}{\bar{\Delta}}$	$\frac{\Delta_N}{\bar{\Delta}}$	$\frac{\Delta_1}{\bar{\Delta}}$	$\frac{\Delta_N}{\Delta}$	$\frac{\Delta_1}{\bar{\Delta}}$	$\frac{\Delta_N}{\bar{\Delta}}$	$\frac{\Delta_1}{\bar{\Delta}}$	$\frac{\Delta_N}{\bar{\Delta}}$	$\frac{\Delta_1}{\bar{\Delta}}$	$\frac{\Delta_N}{\bar{\Delta}}$	$\frac{\Delta_1}{\bar{\Delta}}$	$\frac{\Delta_N}{\bar{\Delta}}$	$\frac{\Delta_1}{\bar{\Delta}}$	$\frac{\Delta_N}{\bar{\Delta}}$
2	1.176	.824	1.111	.889	1.053	.947	1.0	1.0	.952	1.048	.909	1.091	.870	1.130	.833	1.167	.800	1.200
3	1.370	.671	1.230	.787	1.107	.897			.906	1.097	.824	1.187	.752	1.271	.688	1.349	.632	1.421
4	1.579	.542	1.355	.694	1.163	.848			.862	1.147	.745	1.288	.647	1.420	.563	1.545	.492	1.662
5	1.803	.433	1.487	.609	1.221	.801			.819	1.199	.672	1.393	.553	1.579	.457	1.755	.379	1.919
6	2.040	.343	1.626	.533	1.281	.756			.778	1.252	.604	1.504	.470	1.746	.368	1.977	.289	2.192
7	2.288	.269	1.772	.464	1.342	.713			.738	1.307	.542	1.618	.398	1.922	.293	2.210	.218	2.487
8	2.547	.210	1.923	.403	1.405	.672			.700	1.363	.485	1.737	.335	2.104	.233	2.452	.162	2.775
9	2.814	.162	2.079	.349	1.469	.632			.663	1.421	.433	1.861	.281	2.293	.183	2.702	.120	3.080
10	3.087	.125	2.241	.301	1.535	.595			.627	1.479	.385	1.988	.235	2.488	.143	2.959	.088	3.392
12	3.651	.072	2.577	.221	1.672	.525			.561	1.601	.303	2.253	.161	2.893	.086	3.490	.047	4.031
14	4.229	.041	2.929	.161	1.815	.461			.500	1.728	.237	2.530	.109	3.315	.051	4.036	.024	4.683
16	4.816	.023	3.293	.116	1.964	.404			.445	1.859	.183	2.819	.073	3.749	.030	4.592	.012	5.341
18	5.409	.013	3.666	.083	2.118	.353			.395	1.995	.140	3.117	.048	4.191	.017	5.155	.006	6.004
20	6.003	.005	4.037	.046	2.246	.273			.312	2.102	.089	3.407	.024	4.634	.007	5.719	.002	6.668



$$\Delta = \frac{\text{TOTAL LENGTH}}{\text{NUMBER INCREMENTS}} = \frac{L}{N} \qquad \Delta_{I+1} = \Delta_I \cdot \text{EXPAND}$$

$$\frac{\Delta_1}{\Delta} = \frac{N\,(1 - \text{EXPAND})}{(1 - \text{EXPAND}^N)} \qquad \frac{\Delta_N}{\Delta} = \frac{N\left(1 - \frac{1}{\text{EXPAND}}\right)}{\left[1 - \left(\frac{1}{\text{EXPAND}}\right)^N\right]}$$

Figure 16. Nodal Spacing for Various Expansion Factors

X1	=	x coordinate of first node N1 (length)
Y1	=	y coordinate of first node N1 (length)
Z1	=	z coordinate of first node N1 (length)
XN	=	x coordinate of last node NN (length). Leave blank if a single node is entered.
YN	=	y coordinate of last node NN (length)
ZN	=	z coordinate of last node NN (length)
EXPAND	=	Factor by which the distance between adjacent nodes changes. For example, if the distance between the first two nodes is Δ_1, the distance between the second and third nodes is $\Delta_2 = \Delta_1 \cdot$ EXPAND. A summary of relative spacing between the first two nodes and the last two nodes is given in Figure 16.
IX	=	1 will restrain all nodes (N1 . . NN) in the x direction. No restraint if left blank.
IY	=	1 will restrain nodes in y direction.
IZ	=	1 will restrain nodes in z direction.

Rod Shape Node Cards -- The rod geometry descriptions are given in Figure 17. The rod is always generated in a vertical position about the z axix. When viewed from the positive z direction, the nodes are numbered consecutively counterclockwise, inner to outer and downward. Only one-half the rod is generated as shown, and restraints are provided normal to the plane symmetry. The rotation of the rod for oblique impact is obtained with a Scale/Shift/Rotate Card.

- Rod Node Identification Card (I5) -

2	=	Identification number for rod geometry.

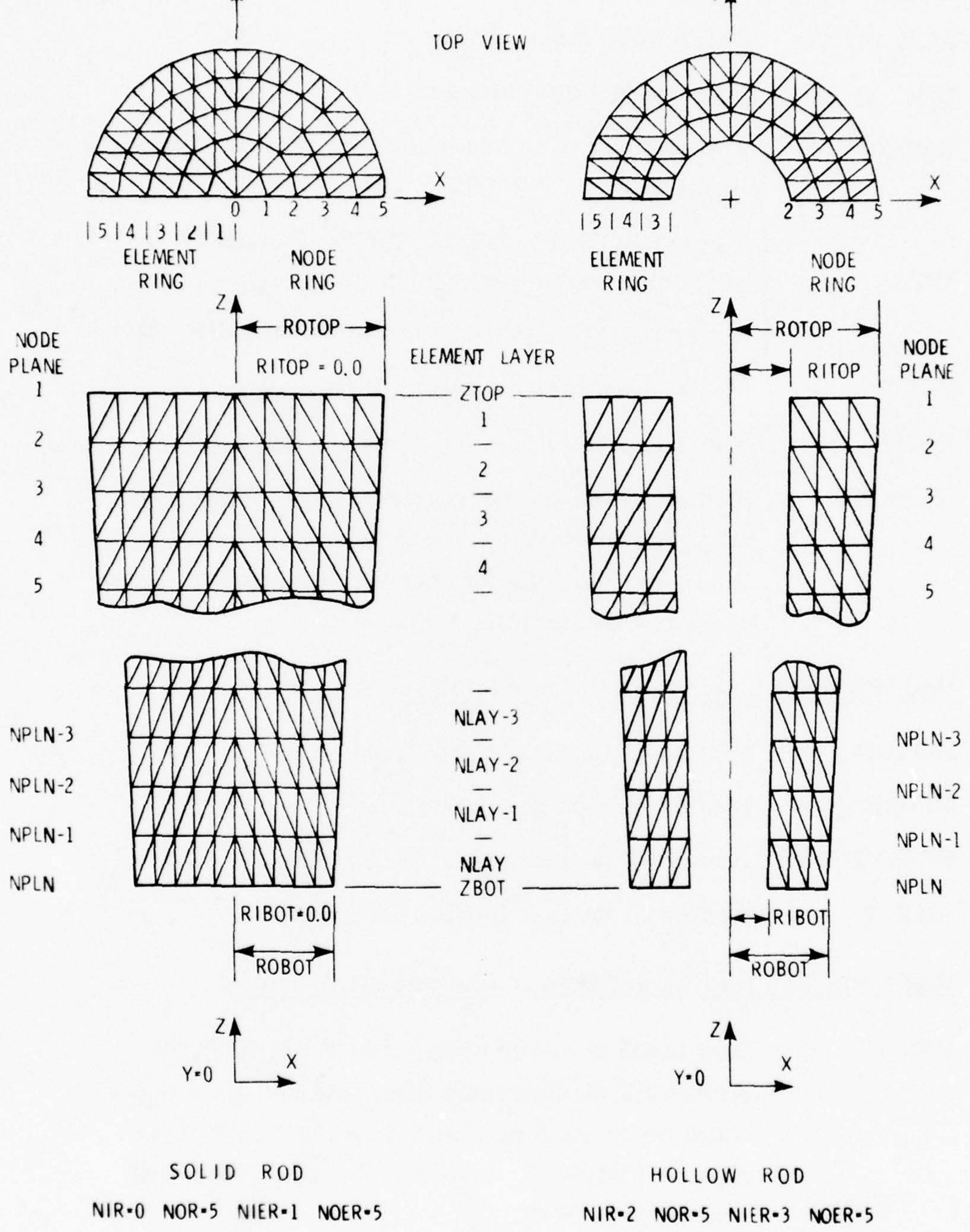


Figure 17. Geometry for Rod Shapes

- Rod Node Description Card (4I5, 3F10.0) -

NOR	=	Outer node ring number.
NIR	=	Inner node ring number.
NPLN	=	The number of cross-sectional planes of nodes in the rod.
IRAD	=	0 gives ROTOP, RITOP, ROBOT, RIBOT input option for uniform radial spacing. 1 gives RT(I), RB(I) input option (I = NIR, NOR).
ZTOP	=	The z coordinate of the top of the rod.
ZBOT	=	The z coordinate at the bottom of the rod.
EXPAND	=	Factor by which the vertical distance between adjacent node changes. Factor applies from top to bottom. Same as described for the Line of Nodes Description Card.

- Rod Node Radii Card for IRAD = 0 (4F10.0) -

ROTOP	=	Radius of top outer node ring of rod.
RITOP	=	Radius of top inner node ring of rod.
ROBOT	=	Radius of bottom outer node ring of rod.
RIBOT	=	Radius of bottom inner node ring of rod.

- Rod Node Top Radii Card (s) for IRAD = 1 (8F10.0) -

RT(I)	=	Top radii for each node ring, I = NIR, NOR. Array RT is currently dimensioned for a maximum outer ring number of 16 (NOR. LE. 16). Note: If NIR = 0, then RT(0) internally set to 0 and I = 1, NOR.

- Rod Node Bottom Radii Card (s) for IRAD = 1 (8F10.0) -

RB(I)	=	Bottom radii for each node ring, I = NIR, NOR. Array RB is currently dimensioned for a maximum outer ring number of 16 (NOR. LE. 16). Note: If NIR = 0, then RB (0) internally set to 0 and I = 1, NOR.

Note: If it is not possible to describe the node geometry of the rod with a single shape, it is possible to use multiple shapes to form a single rod. The nodes must be numbered consecutively, and the radii and the number of node rings must be the same for the individual rod shapes at their interface. Also, ZBOT and ZTOP for adjoining rods should not be identical. ZTOP for the lower rod should be less than ZBOT for the upper rod by the desired node spacing in the z direction.

Nose Shape Node Cards -- The nose geometry descriptions are given in Figure 18. The nose shape is always generated in a vertical position (pointed downward) about the z axis. When viewed from the positive z direction, the nodes are numbered consecutively, counterclockwise, downward and inner to outer. Only one-half the nose shape is generated as shown, and restraints are provided normal to the plane of symmetry. The node geometry for the flat plate at the rod interface is not generated with the nose generator and must therefore be generated with the rod generator. The number of rings must be identical for the rod and the nose.

- Nose Node Identification Card (I5) -

3	=	Identification number of nose geometry.

- Nose Node Description Card (4I5, 4F10.0) -

NOR	=	Outer node ring number.
NIR	=	Inner node ring number.

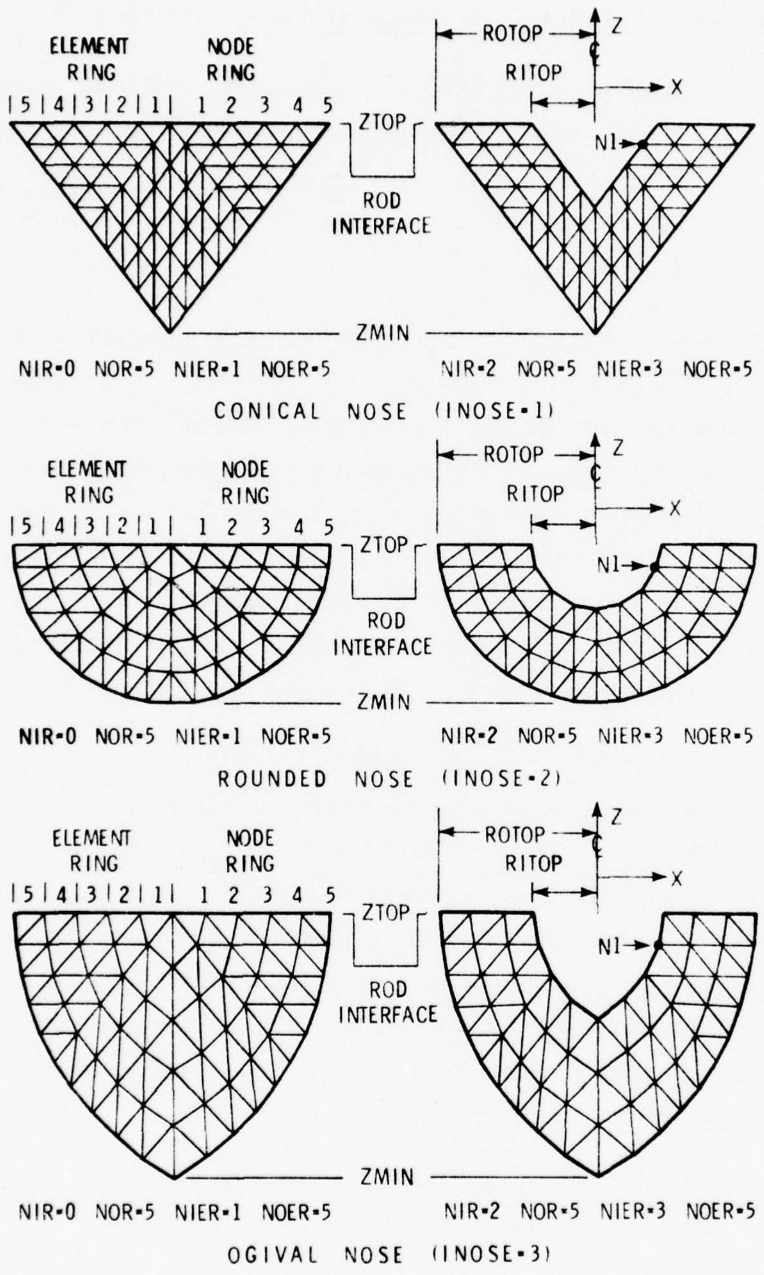


Figure 18. Geometry for Various Nose Shapes

INOSE	=	1 identifies a conical nose shape. 2 identifies a rounded nose shape. If the length of the nose is equal to the radius, a spherical shape is generated. When the length is not equal to the radius, the axial coordinates are scaled and the various radii are not changed. 3 identified a tangent ogival nose shape.
IRAD	=	0 gives ROTOP, RITOP, ZTOP, ZMIN input option for uniform spacing. 1 gives RT(I), ZM(I), ZTOP input option (I = NIR, NOR)
ROTOP	=	Top outer node radius of nose.
RITOP	=	Top inner node radius of nose.
ZTOP	=	The z coordinate at the top of the nose. This is identical to ZBOT for the rod shape at the rod interface.
ZMIN	=	The z coordinate at the tip of the nose.

- <u>Nose Node Top Radii Card(s) for IRAD = 1 (8F10.0)</u> -

RT(I)	=	Top radius for each node ring, I = NIR, NOR. Array RT is currently dimensioned for a maximum outer ring number of 16 (NOR. LE. 16). Note: If NIR = 0, then RT(0) internally set to 0 and I = 1, NOR.

- <u>Nose Node ZMIN Card(s) for IRAD = 1 (8F10.0)</u> -

ZM(I)	=	Minimum coordinates for each node ring, I = NIR, NOR. Array ZM is currently dimensioned for a maximum outer ring number of 16 (NOR. LE 16). <u>Note</u>: If NIR = 0, then ZM(0) internally set to ZTOP and I = 1, NOR.

Flat-Plate Shape Node Cards -- The flat-plate descriptions are given in Figure 19. In all cases, the lines connecting the adjacent corner nodes are parallel to one of the three primary axes. The nodes are generated in rows parallel to the x axis and are numbered consecutively within each row in the direction of the increasing x axis. This generation continues row by row in the positive y direction, plane by plane in the negative z direction.

- Flat-Plate Node Identification Card (I5) -

4	=	Identification number for flat-plate geometry.

- Flat-Plate Node Description Card (6I5, 5F10.0) -

NX	=	The number of nodes in the x direction; in Figure 19, NX = 13.
NY	=	The number of nodes in the y direction; in Figure 19, NY = 7.
NZ	=	The number of nodes in the z direction; in Figure 19, NZ = 5.
NXEND	=	The number of nodes in the x direction in each of the two variable x spacing regions. The spacing is determined by X-EXPAND and the fractional length by X-PART. In Figure 19, NXEND = 4. Depending on whether NX is odd or even, NXEND can have a maximum value of either (NX +1)/2 or NX/2, respectively, unless the special option discussed in X-PART is used. The remaining middle x region (if any) is uniformly spaced.
NYEND	=	The number of nodes in the y direction in the variable y spacing region. Spacing is determined by Y-EXPAND and the fractional length

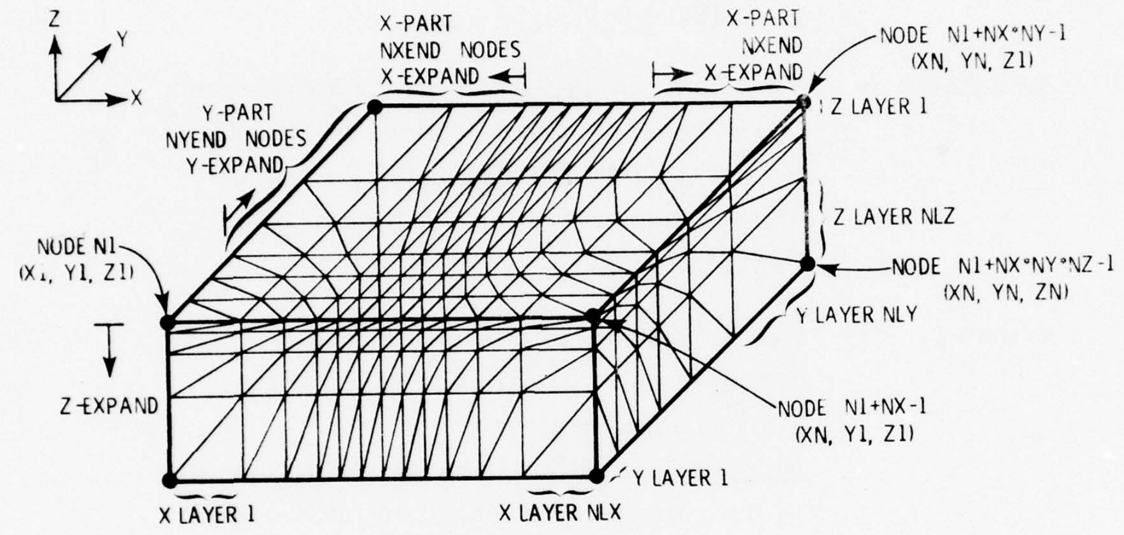


Figure 19. Geometry for Flat-Plate Shapes

		by Y-PART. In Figure 19, NYEND = 4. NYEND can have a maximum value of NY. The remaining y region (if any) is uniformly spaced.
IY	=	1 gives restraint in the y direction only if y = 0 (Y1 = 0. 0).
X-EXPAND	=	Factor by which the x distance between adjacent nodes changes outward to the ends for each X-PART variable spacing region. Same as described for the Line of Nodes Description Card.
X-PART	=	Fractional part of the total x length of the flat plate occupied by each of the variable x spacing regions. <u>Note:</u> If X-PART = 0. 0, the entire spacing in the x direction is uniform. If X-PART = 1. 0 and NXEND = NX, the spacing on the positive x direction is variable for entire x length (X1 to XN)
Y-EXPAND	=	Factor by which the y distance between adjacent nodes changes in the increasing y direction for the Y-PART variable spacing region.
Y - PART	=	Fractional part of the total y length of the flat plate occupied by the variable y spacing region.
Z-EXPAND	=	Factor by which the z distance between adjacent nodes changes in the decreasing z direction from Z1 to ZN.

- Flate-Plate Node Size Card (6F 10.0) -

X1	=	The minimum x coordinate of the plate shape
Y1	=	The minimum y coordinate of the plate shape
Z1	=	The maximum z coordinate of the plate shape
XN	=	The maximum x coordinate of the plate shape
YN	=	The maximum y coordinate of the plate shape
ZN	=	The minimum z coordinate of the plate shape

Sphere Shape Node Cards -- The cross-section of the bottom spherical shape is identical to that shown for the rounded nose shape of Figure 18. The top one-half cross-section is initially geometrically summetric to the bottom. Only one-half of the top and bottom halves are generated, and restraints are provided normal to the vertical plane of symmetry. The sphere is generated with the nodes numbered as two rounded circular noses having an interface between. The top nose is generated first; viewed from the positive z direction, this generation is counterclockwise, upwards and inner to outer. The bottom nose is generated with the interface included with each spherical shell; this generation viewed from the positive z direction is counterclockwise, downwards and inner to outer. A summary of the number of nodes included in the various shapes is given in Figure 20.

- Sphere Node Identification Card (I5) -

5	=	Identification number for sphere geometry.

- Sphere Node Description Card (2I5, 5X, 1I5, 3F10 0)-

NOR	=	Outer node ring number.
NIR	=	Inner node ring number.
IRAD	=	0 gives RO, RI, ZCG input option for uniform spacing. 1 gives RT(I), ZCG input option (I = NIR, NOR).

NUMBER OF RINGS	ROD *		NOSE *		SPHERE *	
	NODES PER PLANE	ELEMENTS PER LAYER	NODES **	ELEMENTS	NODES	ELEMENTS
0	1	-	0	-	1	-
1	6	12	6	12	18	24
2	15	48	30	96	75	192
3	28	108	84	324	196	648
4	45	192	180	768	405	1536
5	66	300	330	1500	726	3000
6	91	432	546	2592	1183	5184
7	120	588	840	4116	1800	8232
8	153	768	1224	6144	2601	12288
9	190	972	1710	8748	3610	17496
10	231	1200	2310	12000	4851	24000
11	276	1452	3036	15972	6348	31944
12	325	1728	3900	20736	8125	41472
13	378	2028	4914	26364	10206	52728
14	435	2352	6090	32928	12615	65856
15	496	2700	7440	40500	15376	81000
16	561	3072	8976	49152	18513	98304

* ONLY ONE-HALF GENERATED

** DOES NOT INCLUDE NODES AT ROD INTERFACE

Figure 20. Summary of Nodes and Elements in Various Shapes

RO	=	Radius of outer node ring.
RI	=	Radius of inner node ring.
ZCG	=	The z coordinate of the center of the sphere.

- Sphere Node Radii Card(s) for IRAD = 1 (8F10.0) -

RT(I)	=	Radii for each node ring, I = NIR, NOR. Array RT is currently dimensioned for a maximum outer ring number of 16 (NOR. LE. 16). Note: If NIR = 0, then RT(0) internally set to 0 and I = 1, NOR.

c. Input Data for Element Geometry

A summary of input data for element geometry is given in Figure 15. Elements may be input as a series of individual or composite elements and/or special shapes of rods, nose geometries, flat plates, and/or spheres. The elements must be assembled in a manner consistent with the previously generated nodal geometry. There is no limit to the number of shapes included for the projectile or the target.

Series of Composite Elements Card -- Element cards for each Series of Composite Elements are supplied as needed by the user. For this discussion it will be assumed that the elements are entered as a series of composite brick elements, each containing six individual elements as shown in Figure 21. Following this immediate discussion will be an example and instructions for generating a series of individual elements.

- Series of Composite Elements Description Card (12I5) -

1	=	Identification number for a series of composite elements.

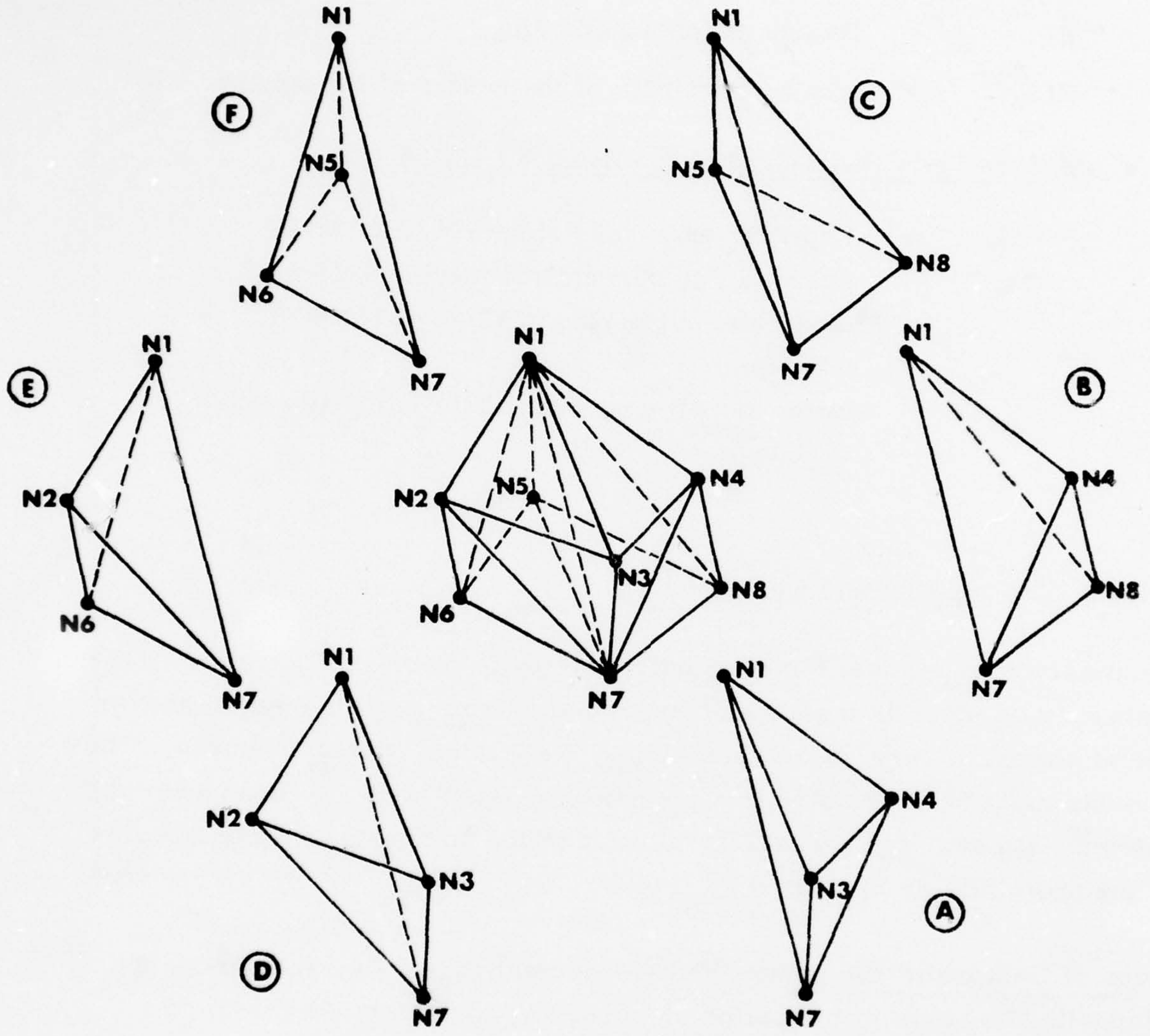


Figure 21. Arrangement of Six Tetrahedra into a Composite Brick Element

NCOMP	=	Number of composite elements in the series.
MATL	=	Material number (1, 2, 3, 4, or 5) of the elements. If left blank, the material number from the previous element data card will be used.
N1-N8	=	Node numbers of the first composite brick element as shown in Figure 21. Nodes N1, N2, N3, N4 and nodes N5, N6, N7, N8 are counterclockwise when looking from N1 to N5.
INC	=	The node number increment added to the node numbers of the previous composite brick element for the next composite brick element.

An example of input data for composite brick elements is shown in Figure 22. In the upper left it can be seen that there are four rows of nodes (1 to 4, 5 to 8, 9 to 12, 13 to 16) which are arranged to contain three composite brick elements. If the first element is numbered 1, then the first composite brick contains elements 1 to 6, the second contains 7 to 12, and the third contains 13 to 18. The first composite brick is defined by nodes N1 = 1, N2 = 5, N3 = 9, N4 = 13, N5 = 2, N6 = 6, N7 = 10, and N8 = 14. Note that N1 to N4 and N5 to N8 are counter-clockwise when looking from N1 to N5. The six individual elements are generated according to the arrangement and order (A, B, C, D, E, F) shown in Figure 21. The node numbers for each successive brick are simply INC = 1 greater than those of previous brick. For the second brick for instance, N1 = 1 + 1 = 2, N2 = 5 + 1 = 6, N3 = 9 + 1 = 10, N4 = 13 + 1 = 14, N5 = 2 + 1 = 3, N6 = 6 + 1 = 7, N7 = 10 + 1 = 11, and N8 = 14 + 1 = 15.

It is possible to generate a series of individual tetrahedron elements by letting N1 to N4 be the nodes of the first element, where N1, N2, and N3 are counterclockwise when viewed from N4. This option is exercised when N5 to N8 are

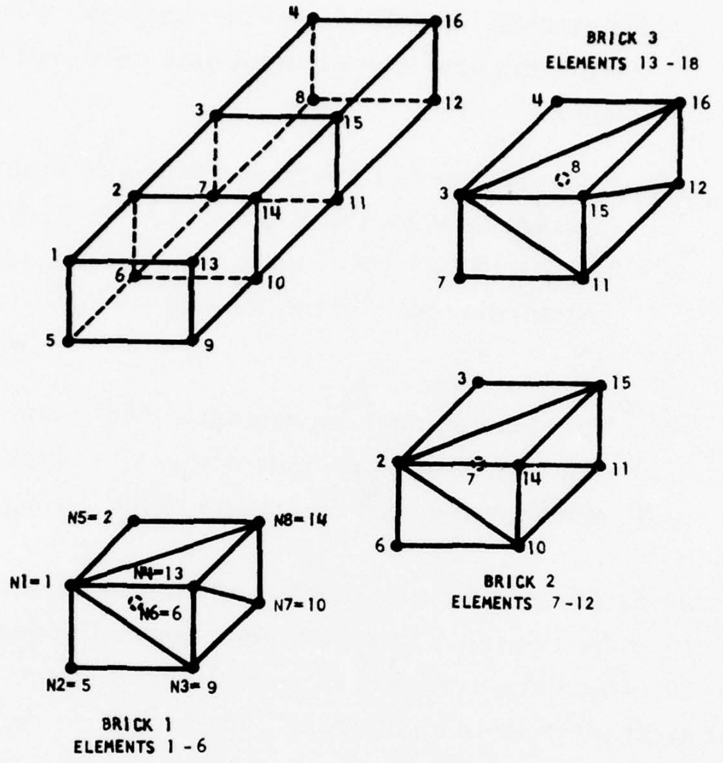


Figure 22. Node Element Input Data Example

left blank. It is also possible to generate a series of composite wedge elements, each containing three individual tetrahedron elements. The three elements in a composite wedge element are numbered consecutively. If N2 and N6 are left blank, the first three tetrahedron elements (A, B, C) are defined by nodes N1, N3, N4, N5, N7 and N8 as shown in Figure 21. Likewise, if N4 and N8 are left blank, the first three elements (D, E, F) are defined by nodes N1, N2, N3, N5, N6, and N7.

Rod Shape Element Cards -- Elements are generated for the rod shapes illustrated in Figure 17 and described by the Rod Shape Node Cards. The elements are numbered consecutively and are generated in layers of composite brick elements beginning with top layer 1 and ending at bottom layer NLAY. The entire first layer of elements is generated before the second layer, etc., and the composite brick elements of each layer are generated in a counterclockwise manner for each ring of elements from the inner to the outer ring.

- Rod Element Identification Card (I5) -

2	=	Identification number for rod geometry.

- Rod Element Description Card (6I5)-

NOER	=	Outer element ring number.
NIER	=	Inner element ring number.
NLAY	=	The number of layers of elements in the rod. The total number of elements in a rod shape shown in Figure 17 is dependent on the number of layers and the number elements per layer. The number elements per layer is dependent on the inner and outer element ring numbers. [For example: If NLAY = 10, NIER = 2 and NOER = 5, the total number of

		elements can be determined through use of Figure 20. The number of elements per layer for the solid rod of NOER = 5 is 300, and of NIER = 2 is 48. Therefore the total of elements for the hollow cylinder is 10 • (300-48) = 2520]
N1	=	The number of the lowest numbered rod node. For the solid rod, this is the centerline node on the top end of the rod. For the hollow rod, this is the innermost clockwise node on the top end of the rod when viewed from the top.
IMAT	=	0 gives MATL input option. Material number of all element rings uniformly set to MATL. 1 gives M(I) input option (I = NIER, NOER). Material numbers for each element ring must be input.
MATL	=	Material number (1. 2, 3, 4 or 5) of the uniform material rod.

- Rod Element Card for IMAT = 1 (16I5) -

M(I)	=	Material number for each element ring, I = NIER, NOER. Array M is currently dimensioned for a maximum outer ring number of 16 (NOER. LE. 16).

Nose Element Cards -- Elements are generated for the nose shapes illustrated in Figure 18 and described by the Nose Shape Node Cards. The elements are numbered consecutively and are generated in shells of composite brick elements beginning with the innermost shell and ending with the outermost shell.

The entire first shell of elements is generated before the second shell, etc., and the composite brick elements of each shell are generated in a counter-clockwise manner for each ring of elements from the top to the bottom of each shell.

- Nose Element Identification Card (I5)-

3	=	Identification number for nose geometry.

- Nose Element Description Card (2I5, 5X, 3I5)-

NOER	=	Outer element ring number.
NIER	=	Inner element ring number.
N1	=	The number of the lowest numbered nose node. (The nose does not include interface nodes.)
IMAT	=	0 gives MATL input option. Material number of all element rings uniformly set to MATL. 1 gives M(I) input option (I = NIER, NOER). Material number for each element ring must be input.
MATL	=	Material number (1, 2, 3, 4, or 5) of the uniform material nose.

- Nose Element Material Card for IMAT = 1 (16I5)-

M(I)	=	Material number for each element ring, I = NIER, NOER. Array M is currently dimensioned for a maximum outer ring number of 16 (NOER.LE.16).

Flat-Plate Shape Element Cards -- The elements are generated for the flat-plate illustrated in Figure 19 and described by the Flat-Plate Shape Node Cards. The elements are numbered consecutively and are generated in rows of composite brick elements. The rows of elements go in the direction of the increasing x axis. Each row of elements is to the positive y direction of the preceding row. After a layer is complete, the next lower layer is generated in a similar manner.

- Flat-Plate Element Identification Card (I5)-

4	=	Identification number for flat-plate geometry.

- Flat-Plate Element Description Card (4I5, 5X, I5)-

NLX	=	Number of layers of composite brick elements in the x direction The total number of nodes along the x direction must be NLX + 1. In Figure 19, NLX = 12.
NLY	=	Number of layers of composite brick elements in the y direction. The total number of nodes along the y direction must be NLY + 1. In Figure 19, NLY = 6.
NLZ	=	Number of layers of composite brick elements in the z direction. In Figure 19, NLZ = 4
N1	=	The node number of the corner node shown in Figure 19.
MATL	=	Material number (1, 2, 3, 4 or 5) of the flat-plate.

Sphere Shape Element Cards -- Elements are generated for a sphere, the bottom half cross section of which is identical to the rounded nose shown in Figure 18 and described by the Sphere Shape Nose Cards. When viewed from the top, the elements are consecutively numbered counterclockwise, upwards and outwards for the top one-half and then counterclockwise, downwards and outwards for the bottom one-half.

- Sphere Element Identification Card (I5)-

5	=	Identification number for sphere shape.

- Sphere Element Description Card (2I5, 5X, 3I5)-

NOER	=	Outer element ring number.
NIER	=	Inner element ring number.
N1	=	The number of the lowest numbered sphere node
IMAT	=	0 gives MATL input option. Material number of all element rings uniformly set to MATL. 1 gives M(I) input option (I = NIER, NOER). Material number for each element ring must be input.
MATL	=	Material number (1, 2, 3, 4 or 5) of the uniform material sphere.

- Sphere Element Material Card for IMAT = 1 (16I5)-

M(I)	=	Material number for each element ring, I = NIER, NOER. Array M is currently dimensioned for a maximum outer ring number of 16 (NOER. LE. 16).

d. Sample Input Data for the Preprocessor

Input data for a typical problem involving oblique impact of a copper rod onto a steel plate are given in Section V.

2. INPUT DATA FOR THE MAIN ROUTINE

The function of the Main Routine is to perform the computations. The Main Routine reads initial conditions from the restart tape which must be generated from a Preprocessor run or a previous Main Routine run. The descriptions which follow are for the data in Figure 23. Consistent units must be used.

- Description Card - A description of the problem provided by the user.

- Identification Card (3I5, 5X, F10.0) -

CASE	=	Case number for run identification
CYCLE	=	The cycle number at which the restart occurs. The cycle numbers for which restart files are written are given in the printed output of the previous run. (Preprocessor or Main Routine)
IPRES	=	0 gives no applied pressures read or applied 1 will use the pressure data which was input in a previous run. 2 will read applied pressures to be used in subsequent computations.

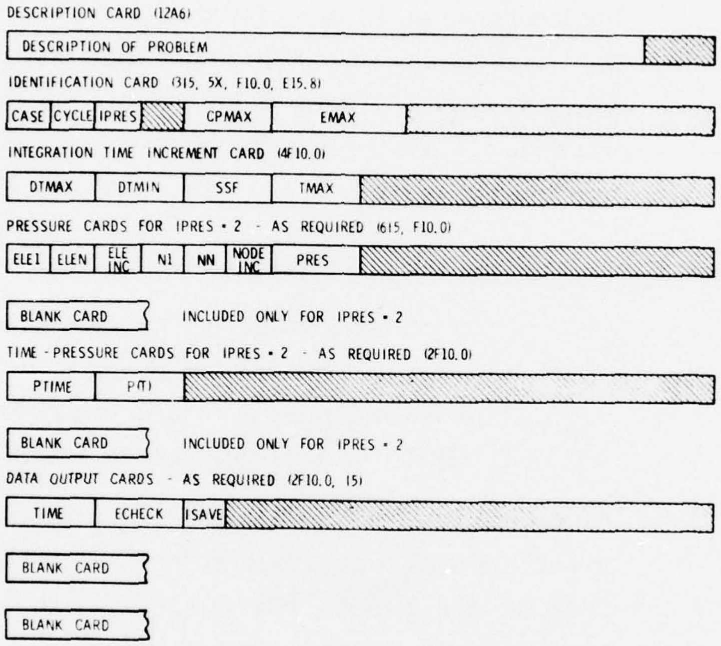

DESCRIPTION CARD (12A6)
DESCRIPTION OF PROBLEM
IDENTIFICATION CARD (3I5, 5X, F10.0, E15.8)
CASE
CYCLE
IPRES
CPMAX
EMAX
INTEGRATION TIME INCREMENT CARD (4F10.0)
DTMAX
DTMIN
SSF
TMAX
PRESSURE CARDS FOR IPRES = 2 - AS REQUIRED (6I5, F10.0)
ELE1
ELEN
ELE INC
N1
NN
NODE INC
PRES
BLANK CARD
INCLUDED ONLY FOR IPRES = 2
TIME-PRESSURE CARDS FOR IPRES = 2 - AS REQUIRED (2F10.0)
PTIME
P(T)
BLANK CARD
INCLUDED ONLY FOR IPRES = 2
DATA OUTPUT CARDS - AS REQUIRED (2F10.0, I5)
TIME
ECHECK
ISAVE
BLANK CARD
BLANK CARD

Figure 23. Main Routine Input Data

CPMAX	=	Central Processor time (hours) at which the results will be written onto the restart tape and the run will stop. This feature will be bypassed if CPMAX = 0.
EMAX	=	The upper limit for total kinetic energy if applied pressures are included (IPRES = 1 or 2). This is used for numerical instability checks. Run will stop if the kinetic energy exceeds EMAX. Leave blank if there are no applied pressures.

- Integration Time Increment Card (4F10. 0)-

DTMAX	=	The maximum integration time increment which will be used for the equations of motion. If Δt from Equation (54) is greater than DTMAX, it will be redefined as Δt = DTMAX.
DTMIN	=	The minimum integration time increment allowed. If Δt from Equation (54) is less than DTMIN, the results will be written onto the restart tape, and the run will stop.
SSF	=	The fraction of the sound speed transit time used for the integration time increment. This is identical to C_t in Equation (54) and must be less than unity.
TMAX	=	The maximum time the problem is allowed to run. This time refers to the dynamic response of the system, not the central processor time (CPMAX) described in the Identification Card. The results at time = TMAX are written onto the restart tape, and the run is discontinued.

- Pressure Cards for IPRES = 2 (6I5, F10.0)- These cards describe the applied pressures and the elements to which they are applied. If other pressures were used previously, they are all deleted, and the only applied pressures which act are those which are input in the current run. End with a blank card.

ELE1	=	The first element. in a series of elements, to which pressure is applied. Must not be less than ELE1 or ELEN from a previous Pressure Card.
ELEN	=	The last element in series of elements. Cannot be less than ELE1.
ELE INC	=	The element number increment between ELE1 and ELEN. If ELE1 = 100, ELEN = 120 and ELE INC = 5, then pressures are applied to elements 100, 105, 110, 120.
N1	=	The node number opposite the triangular face of element ELE1 to which the pressure is applied.
NN	=	The node number opposite the triangular face of element ELEN to which the pressure is applied.
NODE INC	=	The node number increment between N1 and NN. For the elements described under ELE INC (100, 105, 110, 115. 120), if N1 = 200, NN = 208 and NODE INC = 2, then the pressures are applied to the triangular faces opposite nodes, 200, 202, 204, 206, 208, of elements 100, 105, 110, 115, 120.
PRES	=	The pressures which are applied to the triangular faces of the elements described on this card (force/area).

- Time-Pressure Cards for IPRES = 2 (2F10. 0) - These cards allow the applied pressures to be varied as a function of time. A minimum of two cards must be used which span the time from the beginning of the run to TMAX. Program currently dimensioned for a maximum of 50 cards. End with a blank card.

PTIME	=	The time corresponding to P(T). Cards must be input in order of increasing time.
P(T)	=	The factor by which all pressures are multiplied at the corresponding time. Equal to F_t in Equations (48) to (50). Intermediate values are linearly interpolated between values at specified times.

- Data Output Cards (2F10. 0, I5) - These cards are used to specify various forms of output data at selected times, and the last card must be for a time greater than TMAX, even though output will not be provided for that specific time. Recall that output is automatically provided at TMAX and a data output card need not be provided for this time. End run with two blank cards.

TIME	=	Time at which output will be provided.
ECHECK	=	A code which governs the printed output. Two options are provided

(1) If ECHECK is greater than 1000, the individual node and element data will not be printed. Only system data such as centers of gravity, energies, momenta, and net velocities are provided for the projectile, target, and total system.

(2) If ECHECK is less than 1000, the system data and individual node data will be printed. Individual element data will be printed for all elements which have an equivalent plastic strain [Equation (39)] equal to or greater than ECHECK. For example, if ECHECK = 0, all element data will be printed. If ECHECK = 0. 5, only those elements with equivalent plastic strains equal to or greater than 0. 5 will have data printed. If ECHECK = 999, no element data will be printed.

Note: There are some instances when data are printed even though not specified with a data output card. If the minimum time increment, DTMIN, is violated, or the specified central processor time, CPMAX, is exceeded or the maximum run time, TMAX, is achieved, the results are written on the restart tape. The value of ECHECK used for the printed output is that value which is on the following data output card

ISAVE = $\begin{cases} \text{0 will not write results on the restart tape.} \\ \text{1 will write results on the restart tape.} \end{cases}$

3. INPUT DATA FOR THE POSTPROCESSOR

The function of the Postprocessor is to provide plots of the results in the form of geometry and velocity vectors. The Postprocessor reads data from the restart tape which must be generated by a Preprocessor run or a Main Routine run. The descriptions which follow are for the data in Figure 24. Each plot requires a Plot Identification Card and a Plot Limits Card. The three-dimensional plots also require a 3D Perspective Card. The data must be input in groups of two (2D plots) or three (3D plots) cards and the requested cycle numbers must be equal or greater than the previously requested cycle number. Consistent units must be used. End with a blank card.

- Plot Identification Card (3I5, 5X, F10. 0, 5A6) -

TYPE	=	1 gives a geometry plot 2 gives a velocity vector plot
VIEW	=	A code to specify the view requested. Options are given in Figure 26.
CYCLE	=	The cycle number of the plot which is desired. The cycle numbers of the data written on the restart tape are given in the printed output of the Preprocessor and the Main Routine.
VSCALE	=	The velocity which will give a velocity vector which has a length of 1. 0 inch using the scale of the plot. Leave blank for Geometry Plot.
TITLE	=	A title which is written on the plot

- Plot Limits Card (6F10. 0) - This card specifies the portion of the problem which is plotted. Regions beyond those specified are not plotted. For the two-dimensional plots, the vertical axis is 10 inches long and the horizontal axis is as specified.

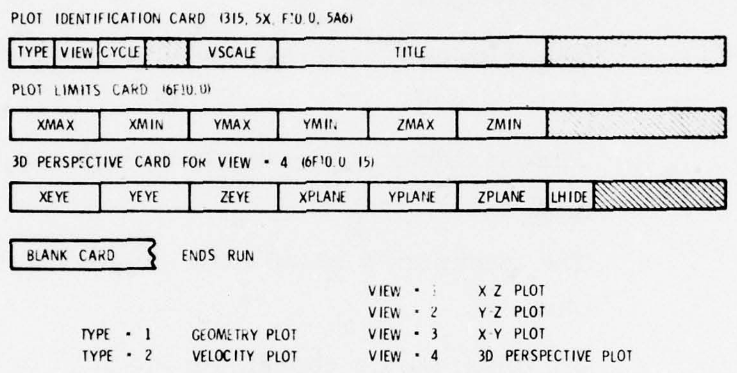


Figure 24. Postprocessor Input Data

The length of the horizontal axis will vary since it has the same scale as the vertical axis. The vertical axes are the z axis (VIEW = 1 and 2) and the y axis (VIEW = 3). For the three-dimensional plots, the axes are scaled such that the entire region is included within the 10-inch vertical axis.

XMAX	=	The maximum x coordinate included in the plot (length)
XMIN	=	The minimum x coordinate included in the plot
YMAX	=	The maximum y coordinate included in the plot
YMIN	=	The minimum y coordinate included in the plot
ZMAX	=	The maximum z coordinate included in the plot
ZMIN	=	The minimum z coordinate included in the plot

- <u>3D Perspective Card for VIEW = 4 (6F10. 0, I5)</u> - This card is included only for the 3D plots.

XEYE	=	
YEYE	=	Coordinates of the observer (length)
ZEYE	=	
XPLANE	=	Coordinates included in the plane on which the results are plotted. The plane is normal to a line from XEYE, YEYE, ZEYE to XPLANE, YPLANE, ZPLANE.
YPLANE	=	
ZPLANE	=	
LHIDE	=	0 will plot all free surfaces (no hidden lines) 1 will plot only surfaces which are visible to the observer

4. OUTPUT DATA

The output data are generally described using the terminology of this report. A summary of the printed output for a Preprocessor run follows:

- All input data are printed exactly as read.
- Material data are printed for the input data.
- All the geometric input data are printed in a form similar to that used to read the data.
- For each node, the initial coordinates, mass are printed if IPRINT = 1 on the Identification Card. The slave nodes are identified by adding "1" before the XYZ restraint. If the XYZ restraint is 100, the node is restrained in the x direction. If the XYZ restraint is 1100, it is a slave node with the same restraint.
- For each element, the four nodes, volume, and material number are printed if IPRINT = 1.
- A summary of the geometry, bandwidth requirements, mass distributions, and initial dynamic conditions is provided. Note: The node and element block sizes must always be equal to those used in the Preprocessor. The number of node blocks can be varied, however, provided there are at least as many as specified under the bandwidth requirements.
- A summary of system data is provided to give centers of gravity, energies, momenta, and net velocities for the initial condition.

For a Main Routine run, the printed output data conforms to the following:

- All input data are printed exactly as read.

- Pressure data are printed in a form similar to that used to read the data

- For each integration time increment, selected data are printed. These consist of cycle number (CYCLE), the time (TIME), the integration time increment for the next cycle (DTNEXT), the element number which governs the time increment (ECRIT), and the total kinetic energy in the system (TOTAL K. E.). If ECRIT = 0 the integration time increment determined from Equation (54) is either greater than DTMAX, or greater than 1. 1 times the integration time increment for the previous cycle.

- Node data are printed at the selected times specified on the Data Output Cards if ECHECK is less than 1000. These data include the node number (NODE), the coordinates (X, Y, Z), the velocities (XDOT, YDOT, ZDOT) and the accelerations (XDD, YDD, ZDD). If a slave node is in contact with the master surface, or if a master node is affected by a slave node, then the printed accelerations are not correct for these nodes.

- Element data may be printed at the selected times specified on the Data Output Cards if ECHECK is less than 1000. Only those elements with an equivalent plastic strain equal to or greater than ECHECK will have their data printed. These data include the element number (ELE), the volumetric strain (DVOL), the equivalent plastic strain (EPBAR) of Equation (39), the Von Mises equivalent stress (VMISES) of Equation (31), the normal deviator stresses (SX, SY, SZ) of Equations (19)to (21), the shear stresses (SXY, SXZ, SYZ) of Equations (22)to (24), the net pressure (P + Q) containing both the hydrostatic pressure of Equation (42) and the artificial viscosity of Equation (45), and the temperature (TEMP) of Equation (41).

- A summary of system data is printed at the selected times specified on the Data Output Cards. Included are centers of gravity, energies, momenta, and net velocities.

For the Postprocessor, all input data are printed exactly as read.

5. INSTRUCTIONS FOR CHANGING PROGRAM DIMENSIONS

It may sometimes be desirable to change the dimensions of the program. The user should review Section III to understand the data structure of the program.

To change the size of the node blocks, the following changes are required:

- Set NBSIZE to the node block size in the data statements in the Preprocessor (PREP), Main Routine (MAIN), and Postprocessor (POST). This must be consistent with the dimension of the COMMON/NODE/arrays (X...MASTER), and the number of node blocks in core, such that

  NBSIZE = NASIZE/NBAND

  where NBAND is the number of node blocks in core and NASIZE is the dimension of the COMMON/NODE/arrays. NBAND and NASIZE can also be altered in the same data statements.

- Dimension INBUFW to (NBSIZE + 2) and RNBUFW to (11•NBSIZE) in COMMON/NBUFW/.

- Dimension INBUFR to (NBSIZE + 2) and RNBUFR to (11•NBSIZE) in COMMON/NBUFR/.

- Note: The node block size cannot be changed during the course of a run.

To change the size of the node arrays, the following changes are required:

- Dimension the COMMON/NODE/arrays (X... MASTER) to the desired size.

- Set NASIZE to the dimensions of the node arrays in the data statements in the Preprocessor (PREP), Main Routine (MAIN), and Postprocessor (POST). Recall that

  NASIZE = NBSIZE • NBAND

  for consistency with the node block sizes.

- Note: The node array size (NASIZE) can be changed during the course of a run provided the node block size (NBSIZE) does not change.

- Note: Program is currently limited to 100 node blocks (NBAND = 100). It can be increased by changing dimensions of NBTBLE and ISLIDE in COMMON/BOOKKP/.

To change the size of the element block, the following changes are required:

- Set LBSIZE to the element block size in the data statements in the Preprocessor (PREP), Main Routine (MAIN) and Postprocessor (POST).

- Dimension the COMMON/ELEMENT/arrays (NODE1... PRES4) to the element block size (LBSIZE)

- Dimension IEBUFW to (6•LBSIZE + 2) and REBUFW to (15•LBSIZE) in COMMON/EBUFW/.

- Dimension IEBUFR to (6•LBSIZE + 2) and REBUFW to (15•LBSIZE) in COMMON/EBUFR/.

- Note: The element block size (LBSIZE) cannot be changed during the course of a run.

6. FILE DESIGNATIONS

The files are defined in the data statements in the Preprocessor (PREP), Main Routine (MAIN) and Postprocessor (POST). The current designations and descriptions are:

NFILER	= 1	One of two internal node files
NFILEW	= 2	One of two internal node files
LFILER	= 3	One of two internal element files
LFILEW	= 4	One of two internal element files
IN	= 5	Input file
IOUT	= 6	Output file
ISFILE	= 8	Internal sliding surface file
ITAPIN	= 9	Restart tape read by MAIN and POST
ITAPOT	= 10	Restart tape generated by PREP and MAIN
ITRIAN	= 11	Triangle data record generated by POST
ITEMP	= 12	Scratch file used by POST

7. CENTRAL PROCESSOR TIME ESTIMATES

For various problems run on a CDC 6600 computer, the EPIC-3 Main Routine has used from 0.006 to 0.009 central processor second per cycle per node.

SECTION V
EXAMPLES

Several examples have been run to verify and demonstrate the capability of the EPIC-3 code. Two wave propagation solutions are shown in Figure 25. The water column example is for impact onto a rigid boundary at 300 meters/second (Mach = 0.2). The dimensionless pressure is defined as $P^* = P/\rho_o C_o V_o$ where P, ρ_o, C_o and V_o represent the pressure, initial density, acousitic velocity, and impact velocity. The pressure is shown at a time shortly before the wave reaches the far end of the column. It can be seen that the numerical EPIC-3 solution agrees with the analytic solution, $P^* = 1.4$ (Reference 7) along the length of the column with only slight differences occurring along the leading edge of the wave.

The other wave propagation example in Figure 25 shows two aluminum bars impacting at 800 meters/second. This example shows the effect of material strength with a higher velocity elastic wave at the leading edge of the shock front and the elastic unloading at the rear. This solution is in good general agreement with that presented in Reference 8.

The next examples demonstrate the capability to represent plastic flow. Figure 26 shows the results of a steel cylinder striking a rigid frictionless surface at impact velocities of 252 and 402 meters/second. These problems were selected since the final lengths could be compared to test data and HEMP simulations in Reference 9. Using a dynamic yield strength of 1.2 gigi-pascal, the computed results are in good agreement with those reported in Reference 9.

The final lengths for the 252-meter/second impact are $0.842L^o$, $0.842L^o$, $0.854L^o$ for the test results, HEMP computational results and EPIC-3 results

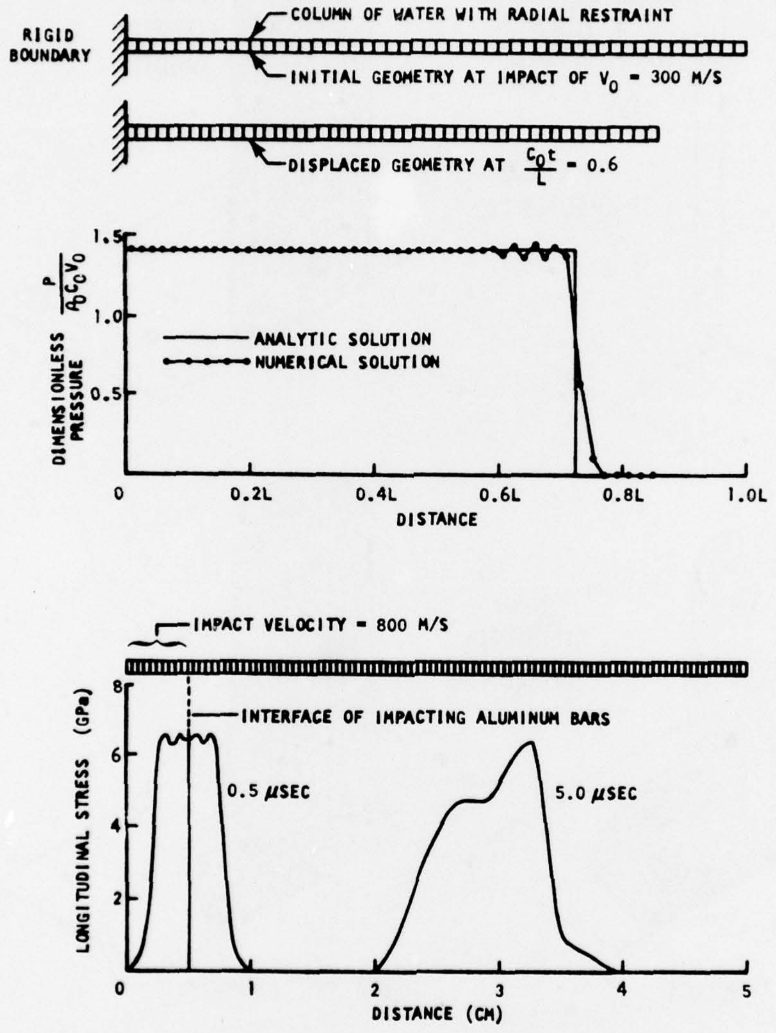


Figure 25. Wave Propagation Examples

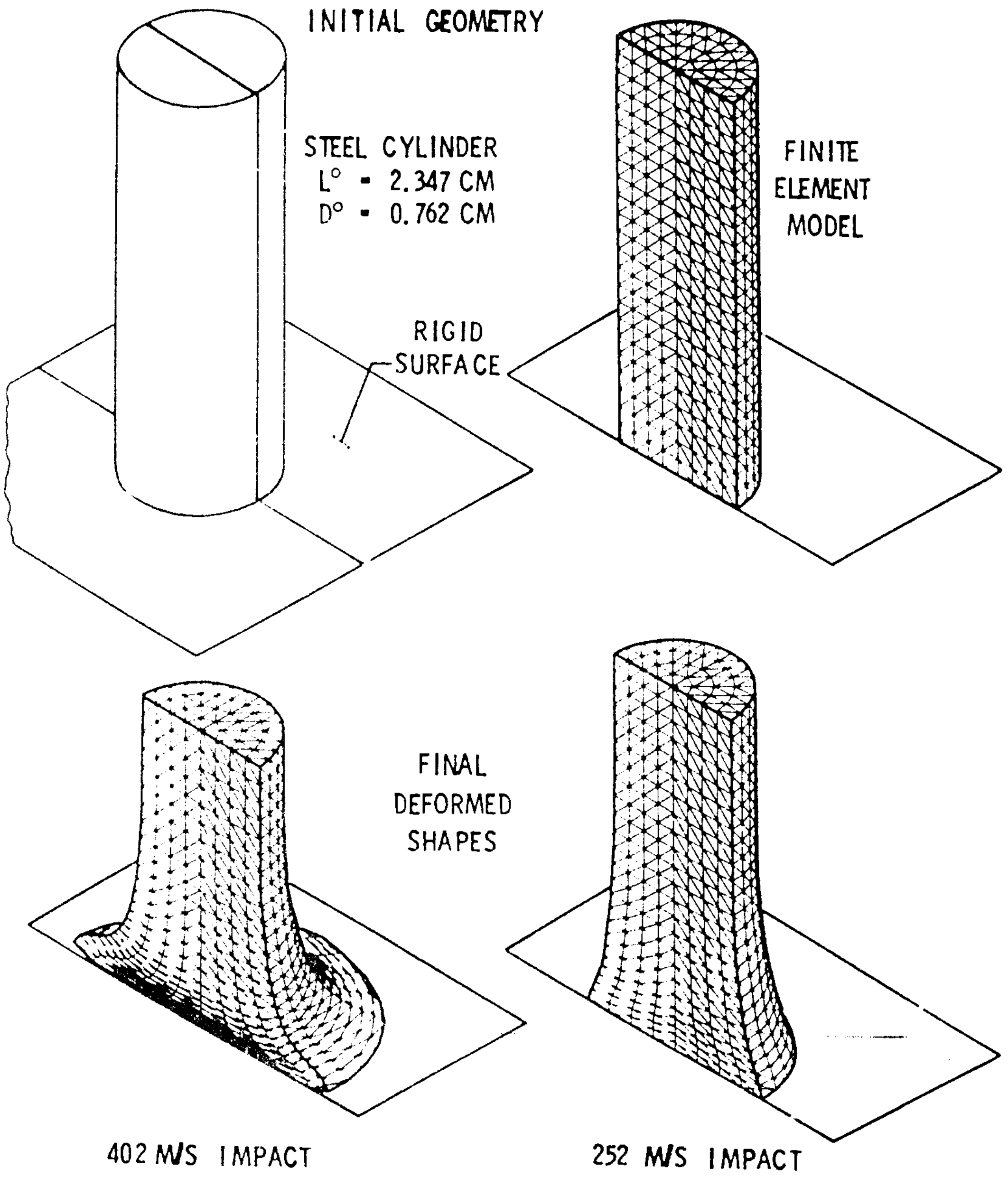


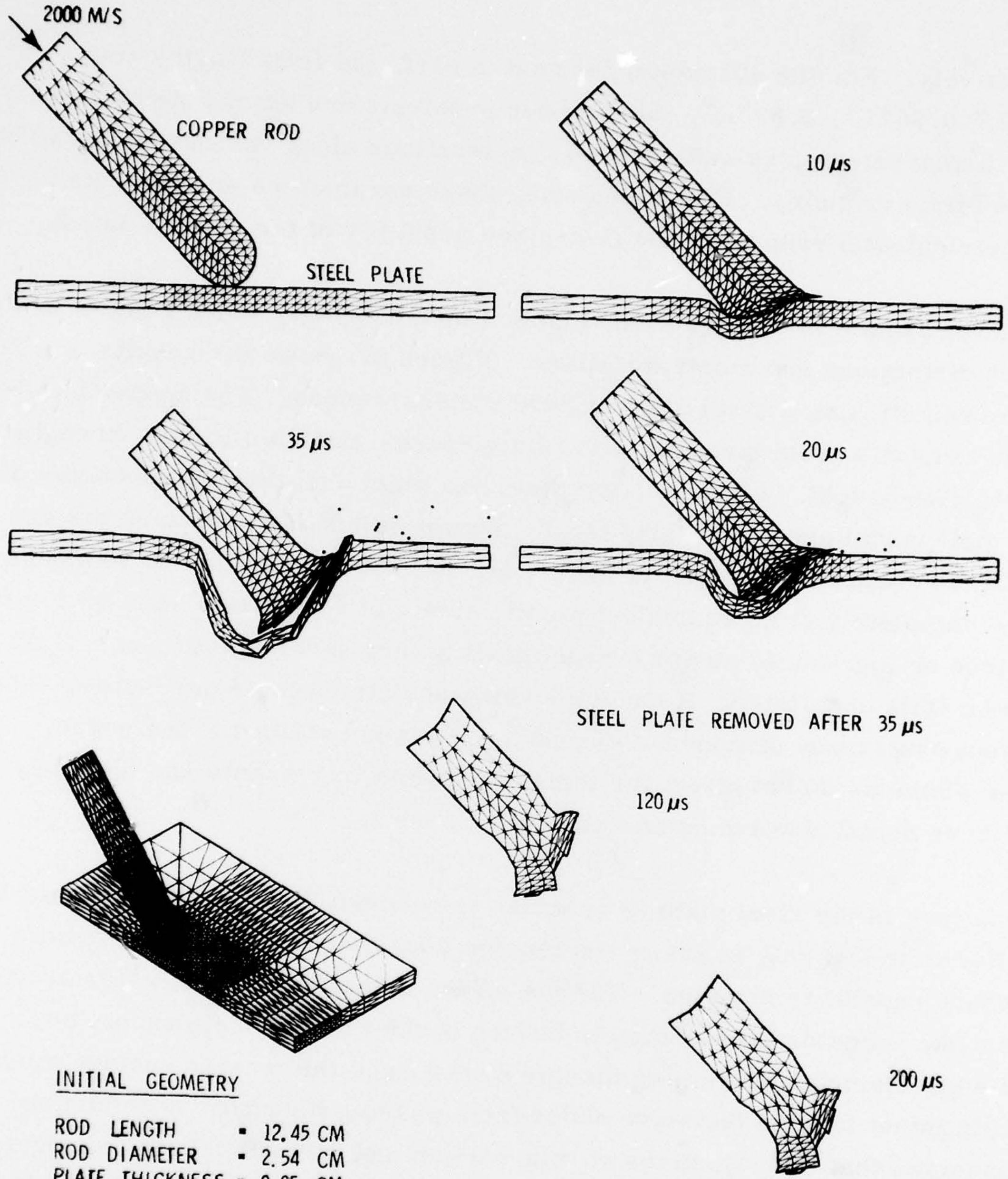
Figure 26. Normal Impact of a Steel Cylinder onto a Rigid Surface at Velocities of 252 and 402 meters/second

respectively. For the 402-meter/second impact, the final lengths are $0.635L^o$ $0.667L^o$, $0.693L^o$. Since these problems are axisymmetric, the nodal displacements, velocities, and accelerations along various radii can be checked for symmetry. For both cases, these parameters are absolutely symmetrical with respect to the five-place accuracy of the printed output.

The next example involves a much higher impact velocity which results in severe distortions and material failure. Figure 27 shows the results of a copper rod striking a steel plate at 2000 meters/second. The copper is assumed to have a yield strength of 0.14 gigi-pascal and an ultimate strength of 0.45 gigi-pascal. Likewise, the steel has yield and ultimate strengths of 1.10 gigi-pascal and 1.40 gigi-pascal. This problem also requires the capability to represent failure of the copper and steel. The copper is assumed to fail completely at an equivalent plastic strain of $\bar{\epsilon}_p = 2.0$. Since this strain is a true or logrithmic strain,it represents a very severe distortion. If an element fails completely, it cannot develop any stresses or pressures. The element essentially disappears except the mass is retained at the nodes. These elements do not affect the integration time increments and they may even have negative volumes and altitudes.

The failure in the steel plate is assumed to occur at $\bar{\epsilon}_p = 0.7$. This material is allowed to fail only in shear and tension such that a positive hydrostatic pressure capability remains. The net effect is that this failed material behaves like a liquid. The complete failure of the rod elements cannot be applied to elements containing master nodes since the master surface must remain intact to keep the slave nodes from passing through. It should be emphasized that the objectives of this work do not include defining accurate material descriptions for these dynamic conditions. This will be a significant area of research during the next few years. By using these approximate values, however, the capability of code can be demonstrated.

The geometry and finite element model are shown in the lower left corner of



INITIAL GEOMETRY

ROD LENGTH	= 12.45 CM
ROD DIAMETER	= 2.54 CM
PLATE THICKNESS	= 0.95 CM
IMPACT ANGLE	= $\pi/4$ RAD

Figure 27. Oblique Impact of a Copper Rod onto a Steel Plate at 2000 meters/second

Figure 27. Only half the problem about the plane of symmetry is included in the model, which contains 2184 nodes and 8568 elements. This will be disignated as the coarse-grid model for subsequent discussions regarding computer requirements.

The deformed geometry at the plane of symmetry is shown at several instances after impact. At 10 microseconds the nose of the rod is beginning to jet outward and at 20 microseconds many of the elements have failed completely. The dots represent concentrated masses at nodes whose associated elements have completely failed. At 35 microseconds the rod has shortened significantly and the plate is removed from the simulation at this time. It can be seen that the rod continues to deform between 35 and 120 microseconds. Thereafter, the remainder of the rod travels in essentially a straight line at a slightly reduced velocity of about 1950 meters/second.

Since three-dimensional computations can require significant computer time, and since the computer time is dependent on the finite element model, a brief discussion of this area is appropriate. The coarse-grid model of Figure 27 has nodal radial spacing equal to one-third the radius and the plate has nodal vertical spacing equal to one-third the thickness of the plate. A fine-grid model was also generated which had reduced nodal spacing equal to about three-fifths that of the coarse grid model. The radial spacing in the rod is one-fifth the radius and the vertical spacing in the plate is one-fifth the thickness. The number of nodes and elements required for the fine grid is increased dramatically to 8784 nodes and 40,400 elements. Generally, the increase in nodes and elements is inversely proportional to the cube of the spacing, or about $(5/3)^3$. In addition, more integration cycles are required since the sound speed transit times in the elements are decreased. Therefore, it would be expected that the computer time required for the fine-grid model would be about $(5/3)^4$ or 7.7 times that required for the coarse-grid model. It should also be noted that the coarse-grid model can contain all nodes in core with a computer memory of about 66K, thus eliminating the need to buffer nodal data between disk files and central memory. The fine-grid model requires slightly less than 66K to contain a bandwidth of data, so the disk files must therefore be used unless the memory is significantly expanded.

A comparison of various parameters for the fine- and coarse-grid models is shown in Figure 28. The fine-grid model was only run to 18.7 microseconds since it required much more computer time. The top of Figure 28 shows the penetration into the plate is approximately equal for both models, with the coarse-grid plate being slightly stiffer and therefore allowing less penetration. The center portion of the same Figure shows the loss of momenta and kinetic energy of the copper rod. It can be seen that there is excellent agreement for the kinetic energy and the horizontal momenta, but the increased plate resistance of the coarse-grid model tends to decrease the vertical momentum of the rod more rapidly. The final comparison shows the large increase in computer time required for the fine-grid model. It can be seen that the fine-grid computation required 12.5 hours of Central Processor time on a CDC 6600 computer for 18.7 microseconds, whereas the coarse grid required only 3.5 hours for 35 microseconds.

Preprocessor input data for the coarse-grid problem are given in Figure 29. Consistent pound-inch-second units are used for all input. Cards 1 and 2 represent the Description Card and Identification Card as described in Figure 13, and Cards 3 through 24 represent the 22 Material Cards. Card 25 is the Projectile Scale/Shift/Rotate Card, and Cards 26 through 33 define nodal geometry for two rod sections and the rounded nose geometry. Card 34 is a blank to end the Projectile Node Input, Card 35 is the Target Scale/

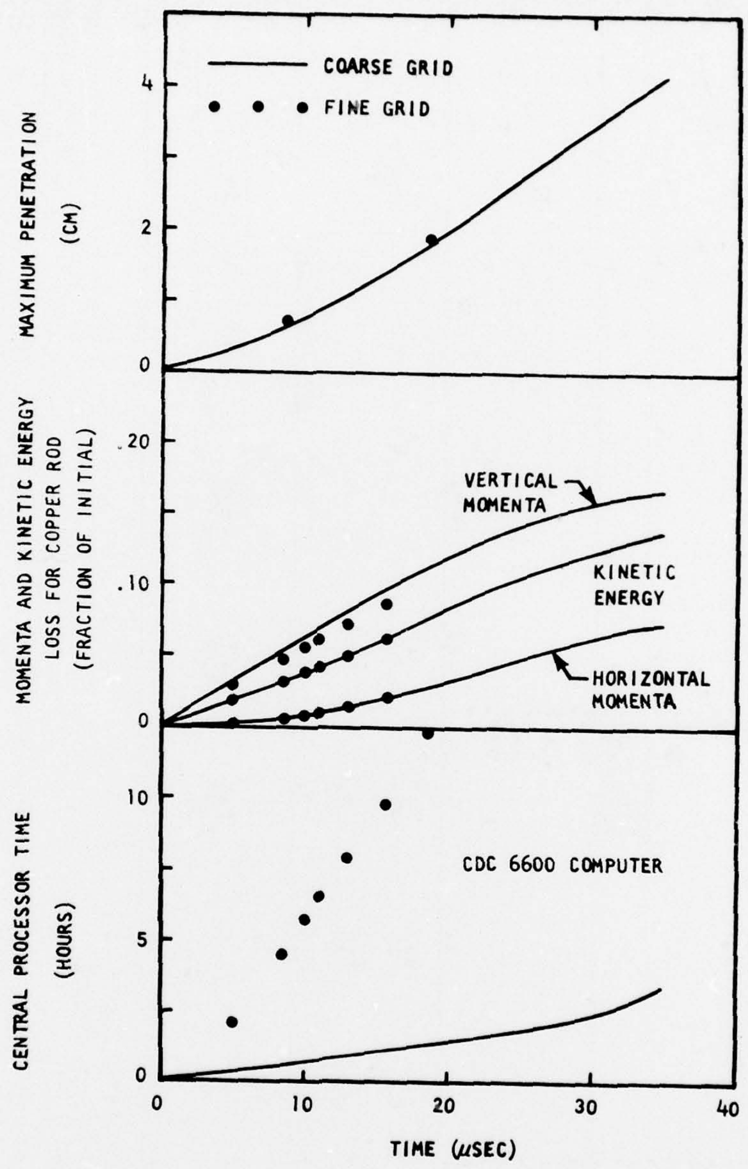


Figure 28. Comparison of Various Parameters for the Coarse and Fine-Grid Models

```
LISTING OF INPUT

OBLIQUE IMPACT OF COPPER ROD ONTO STEEL PLATE AT 2000 M/S - COURSE GRID                                   CARD    1
     4     0     1                                                                                        CARD    2
       .000831                          .000734                                                           CARD    3
       360000.                          410000.                                                           CARD    4
      6380000.                         9300000.                                                           CARD    5
        20300.                          160000.                                                           CARD    6
        65300.                          203000.                                                           CARD    7
            .3                               .3                                                           CARD    8
      9999999.                         9999999.                                                           CARD    9
            0.                               0.                                                           CARD   10
            0.                               0.                                                           CARD   11
            0.                               1.                                                           CARD   12
            1.                               1.                                                           CARD   13
            0.                               0.                                                           CARD   14
     26739000.                        24200000.                                                           CARD   15
     32907000.                        59578000.                                                           CARD   16
     64234000.                        74081000.                                                           CARD   17
          2.00                             1.67                                                           CARD   18
            .2                               .2                                                           CARD   19
           4.0                              4.0                                                           CARD   20
          999.                               .7                                                           CARD   21
          999.                             999.                                                           CARD   22
            2.                             999.                                                           CARD   23
            0.                               0.                                                           CARD   24
       1.          1.         1.         0.          .21        -45.                                      CARD   25
  2                                                                                                       CARD   26
  3    0     8     0         4.9        2.5          .9                                                   CARD   27
      .5          0.          .5         0.                                                               CARD   28
  2                                                                                                       CARD   29
  3    0    13     0         2.3         .5         1.0                                                   CARD   30
      .5          0.          .5         0.                                                               CARD   31
  3                                                                                                       CARD   32
  3    0     2     0          .5         0.          .5          0.                                       CARD   33
                                                                                                          CARD   34
       1.          1.         1.                                                                          CARD   35
  4                                                                                                       CARD   36
 27   14     4     5     5     1        1.5         .31         1.5         .62         1.                CARD   37
   -3.95          0.          0.       3.95        3.95       -.375                                       CARD   38
                                                                                                          CARD   39
  2                                                                                                       CARD   40
  3    1    20     1     0     1                                                                          CARD   41
  3                                                                                                       CARD   42
  3    1         589     0     1                                                                          CARD   43
                                                                                                          CARD   44
  4                                                                                                       CARD   45
 26   13     3   673           3                                                                          CARD   46
                                                                                                          CARD   47
  1  677    19    16     1    27     1  281  672  392     0             0.          .4                    CARD   48
                                                                                                          CARD   49
  55660.          0.     -55660.         0.          0.          0. .00000002                             CARD   50
```

Figure 29. Preprocessor Input Data for Rod-Plate Example

Shift/Rotate Card, and Cards 36 through 38 define the nodal geometry for the flat plate. Card 39 is blank to end the Target Node Input. The Projectile elements are defined by one rod shape and one nose shape with Cards 40 through 43, and ended with blank Card 44. Similarily, Cards 45 and 46 define the target plate elements, ended with blank Card 47. The sliding surfaces are defined with Card 48 and ended with blank Card 49. Finally, the Initial Velocity Card is represented with Card 50.

The coarse-grid problem used nodal arrays dimensioned at NASIZE = 2240 (see Section IV). Since this problem had fewer nodes (2184) it could be core contained, eliminating the need for buffering the nodal data. The node and element block sizes were NBSIZE = LBSIZE = 64 for a total bandwidth capacity of NBAND = 35 node blocks. With these dimensions, the Main Routine requires about 66K central memory on the Eglin Air Force Base 6600 computer. It should be noted that the required bandwidth for this problem is only 11 blocks, so it would be possible to run this problem (with buffering) using reduced nodal arrays dimensioned at NASIZE = 704. This is consistent with NBAND = 11 and NBSIZE = 64.

REFERENCES

1. Johnson, G.R., "EPIC-3, A Computer Program for Elastic-Plastic Impact Calculations in 3 Dimensions," BRL 343, Honeywell, Inc., Defense Systems Division, Hopkins, Minnesota, July 1977.

2. Wilkins, M. L., et al, "A Method for Computer Simulation of Problems in Solid Mechanics and Gas Dynamics in Three Dimensions and Time," UCRL-51574, Rev. 1, Lawrence Livermore Laboratory, Livermore, California, May 1975.

3. Walsh, J.M., et al, "Shock-Wave Compressions of Twenty-Seven Metals. Equation of State of Metals," Physical Review, Volume 108, No. 2, October 1957.

4. Von Neumann, J., and Richtmyer, R.D., "A Method for the Numerical Calculation of Hydrodynamic Shocks," Journal of Applied Physics, Volume 21, 1950.

5. Walsh, R.T., "Finite Difference Methods," Dynamic Response of Materials to Intense Impulsive Loading, Edited by Chou, P.C., and Hopkins, A.K., Air Force Materials Laboratory, 1972.

6. Bertholf, L.D., and Benzley, S.E., "TOODY II, A Computer Program for Two-Dimensional Wave Propagation," SC-RR-68-41, Sandia Laboratories, Albuquerque, New Mexico, November 1968.

7. Huang, Y.C., Hammitt, F.G., and Mitchell, T.M., "Note on Shock-Wave Velocity in High-Speed Liquid Solid Impact," Journal of Applied Physics, Volume 44, April 1973.

8. Wilkins, M. L., "Calculation of Elastic-Plastic Flow," Methods in Computational Physics, Volume 3, Edited by Alder, B., Fernbach, S., and Rotenberg, M., Academic Press, New York 1964.

9. Wilkins, M. L., and Guinan, M.W., "Impact of Cylinders on a Rigid Boundary," Journal of Applied Physics, Volume 44, March 1973.

(The reverse of this page is blank)

INITIAL DISTRIBUTION

SAFLAR	1
HQ USAF/RDQRM	2
HQ USAF/SAMI	1
HQ USAF/XOXFM	1
AFIS/INT	1
HQ AFSC/DLCA	1
HQ AFSC/IGFG	1
ASD/ENFEA	1
AFML/LTM	1
AFWL/NSE	2
AFWL/SUL	1
AUL/LSE 71-249	2
DDC	2
HQ TAC/DRA	1
HQ TAC/INAT	1
6510 ABG/SSD/STOP 238	1
HQ USAFE/DOQ	1
HQ PACAF/DOOFQ	3
COMIPAC/I-232	1
Army Mat Sys Ana Agcy/DRXSY-A	1
Army Mat Sys Ana Agcy/DRSXY-J	1
Redstone Sci Info Cnt	1
USAE Wtrwy Exper Stn	1
Naval Research Lab/Code 2627	1
NAVAIR SYS COMD	1
Naval Surface Wpn Cnt	2
Naval Ordnance Stn	1
Naval Air Test Cnt	2
Naval Ship R&D Cnt	1
USNWC/Code 2333	1
USNWC/Code 326	1
Sandia Lab/Tech Lib	1
AFALT/DL	1
AFATL/DLODL	2
AFATL/DLJ	1
AFIT/LD	1
ASD/ENESH/S. Johns	1
AFML/MXE	1
Harry Diamond Lab/DRXDO-DA	1
Naval Wpns Eval Fac/Wpns Dept	1
AFWL/NSC	1
USNWC/Code 3163	1
USA TRADOC SYS ANA ACT	1
AFATL/DLY	2
AFATL/DLJW	14
ASD/XRP	1
Orlando Tech, Inc	1
Ohio State Univ	1
Honeywell Defense Sys Div	5
US Army Ballistic Rsch Lab DRDAR	5
ADTC/AD	1
ADTC/ADD	1
ADTC/ADDS	1
Command Sciences Corp	1
Sci Rsch Lab/Ford Motor Co	1